TÉCNICO LISBOA

Sideslip Estimation of Formula Student Prototype

André Manuel Pinheiro Antunes

Thesis to obtain the Master of Science Degree in

Mechanical Engineering

Supervisors: Prof. Paulo Jorge Coelho Ramalho Oliveira
Prof. Carlos Baptista Cardeira

Examination Committee

Chairperson: Prof. Paulo Rui Alves Fernandes
Supervisor: Prof. Carlos Baptista Cardeira
Member of the Committee: Prof. João Manuel Ferreira Calado

November 2017

Acknowledgments

First of all, I would like to thank my supervisors, Professor Carlos Cardeira and Professor Paulo Oliveira for their availability, support during this work and all the shared scientific knowledge.

I would also like to thank Eng. Bruno Cardeira, from ISR - Dynamical Systems and Ocean Robotics Lab, for his prompt support and assistance.

I also thank the team of FST Lisboa for all the effort for making the tests happen, as well as all the provided information and background that supported this work.

Last, although no less important, I thank all my family for the encouragement and support. And also my friends for their support, friendship and the cheerful breaks from the work.

Resumo

O objectivo deste trabalho é projectar e afinar um conjunto de estimadores capaz de fornecer o slipangle de um carro de Formula Student, capaz de trabalhar em tempo real, e de alimentar um algoritmo de controlo. O estimador proposto utiliza uma IMU (Inertial Measurment Unit), que contém um acelerómetro um giroscópio e um magnetómetro, um GPS (Global Positioning System), um sensor para o angulo da direcção, o binário produzido pelos motores eléctricos e o modelo dinâmico subjacente ao veículo. É apresentado um modelo exaustivo do carro, assim como modelos para os sensores que são utilizados para testar e afinar os estimadores. São apresentados dois filtros cinemáticos um para a orientação e outro para a velocidade, que alimentam um terceiro estimador que utiliza a dinâmica do veículo. Neste último são comparados dois filtros, um linear e um não-linear, que para os quais são discutidos os seus prós e contras. Todos os estimadores lineares utilizados são baseados na teoria do filtro de Kalman (KF), e no caso do filtro não-linear é utilizados o filtro de Kalman estendido. Os estimadores propostos são depois testados com dados recolhidos de um carro real de Formula Student, e depois implementados num microcontrolador a bordo de um veículo RC (Remote Control) num laboratório. Em ambos os testes, os resultados são validados com sistemas secundários mais precisos.

Palavras-chave: Sideslip, Navegação, Estimação, Sensores, Filtro de Kalman, Filtro de Kalman Estendido

Abstract

The objective of this work is the design and tuning of an estimator architecture that can provide the slipangle of a Formula Student prototype, with the requirement of working in real time, to be able to feed a control algorithm. The proposed architecture uses an Inertial Measurement Unit (IMU) composed of an accelerometer a gyroscope and a magnetometer, a Global Positioning System (GPS), a steering encoder, the torque of the electrical motors and the underlying dynamic model of the vehicle. An exhaustive car model is presented, as well as, sensor models, used to test and tune the estimators. Two kinematic complementary filters are presented for attitude and velocity estimation that feed a third estimator using the vehicle dynamics. For this last one, a linear and a non-linear filter are compared and their pros and cons discussed. All the used estimators are based in Kalman Filter (KF) theory for the linear ones and the Extended Kalman Filter (EKF) for the non-linear. The estimator architecture is then tested with offline data from a real Formula Student prototype and implemented in a RC (Remote Control) vehicle's microcontroller inside the laboratory. Both tests are cross-validated with secondary and more accurate systems.

Keywords: Sideslip, Navigation, Estimation, Sensors, Kalman Filter, Extended Kalman Filter

Contents

List of Tables

List of Figures

Nomenclature

Greek symbols

α_i	Side slip angle of wheel i.
β	Vehicle sideslip angle.
β_i	Vehicle sideslip angle projected on wheel i.
γ	Camber angle.
Γ_i	Generic term denominator differentiation.
δ	Steering angle.
δ_r	Steering angle reading.
δ_{HI}	Magnetic field offset.
Δ_{si}	Equivalent suspension displacement (i=1,...4).
η	Ellipse tilt angle.
λ_r	Reference latitude angle.
μ_r	Resulting friction coefficient.
ν_r	Reference longitude angle.
ρ	Air density.
σ_{SF}	Ellipse scale factor.
ϕ, θ, ψ	Roll, pitch and yaw angle.
ψ_r	Yaw angle reading from sensor.
ω	Generic angular velocity.
$\bar{\omega}_r$	Angular velocities vector reading from sensor.

Roman symbols

A_c	Generic continuous state transition matrix.
A_k	Generic discrete state transition matrix.
A_k^{vle}	Discrete state transition matrix of VLE model.
A_x^{proj}, A_z^{proj}	Vehicle area projection in x and z directions.
B_c	Generic continuous input matrix.
B_k	Generic discrete input matrix.
B_k^{vle}	Discrete input matrix of VLE model.
C_{SI}	Soft iron transformation matrix.
C_{SF}	Ellipse scale matrix.

$C_\alpha, C_{\alpha f}, C_{\alpha r}$	Cornering stiffness, generic, front and rear.
C_d, C_l	Drag and Lift coefficients of the complete vehicle.
C_k	Generic discrete observation matrix.
C_{pi}	Equivalent tyre damping (i=1,...,4).
C_{si}	Equivalent quarter suspension damping (i=1,...,4).
E_j	Ellipse equation coefficients (j=1,...6).
F_D, F_L	Aerodynamic forces, drag and lift.
F_k	Jacobian state transition matrix.
F_r	Resulting force.
F_x, F_y, F_z	Force components.
F_{y_i}, F_{z_i}	Force components at wheel i.
G_i	Ground height input at wheel i.
H_k	Jacobian observation matrix.
I	Identity matrix.
I_ϕ, I_θ, I_ψ	Inertia components to roll, pitch and yaw.
$I_c,\ I_f$	Collector and forward current.
K_{acf}, K_{pcf}	Discrete Kalman gain matrix for the respective filter.
K_{ar_j}	Equivalent anti-roll bar stiffness (j=1,2).
K_j	Discrete Kalman gain entry j.
K_k	Discrete Kalman gain matrix.
K_{pi}	Equivalent tyre stiffness (i=1,...,4).
K_{si}	Equivalent quarter suspension stiffness (i=1,...,4).
M_ϕ, M_θ, M_ψ	Torque components in roll, pitch and yaw.
P_j	Plane equation coefficients (j=1,2,3).
P_k	Discrete error covariance matrix.
P_{vle}, P_{vnle}	Discrete error covariance matrix for the respective filter.
$Q_{acf}, Q_{pcf}, Q_{vle}, Q_{vnle}$	Discrete state weight matrix for the respective filter.
Q_k	Discrete state weight matrix.
R_ϕ, R_θ, R_ψ	Rotation matrices of the roll pitch and yaw.
R_η	Rotation matrix of ellipse tilt angle.
$R_{acf}, R_{pcf}, R_{vle}, R_{vnle}$	Discrete observation weight matrix for the respective filter.
R_{imu}	IMU rotation matrix.
R_k	Discrete observation weight matrix.
R_o	Output resistor.
T	Sampling time.
Vcc_r	Regulated supply voltage.
X_p, Y_p, Z_p	ECEF position coordinates.
X_r, Y_r, Z_r	ECEF reference position coordinates.
Z	Center of gravity vertical position.

a Distance from center of gravity to front axle.

$\bar{\mathbf{a}}_k$ Discrete acceleration vector.

$\bar{\mathbf{a}}_r$ Acceleration reading from sensor.

b Distance from center of gravity to rear axle.

$b_{\dot{\psi}k}$ Discrete yaw rate bias.

$b_{\dot{\phi}}, b_{\dot{\theta}}, b_{\dot{\psi}}$ Bias of gyroscope components.

$b_{a_x}, b_{a_y}, b_{a_z}$ Bias of accelerometer components.

$\bar{\mathbf{b}}_{ak}$ Discrete acceleration bias vector.

c Distance from center of gravity to left wheels.

c_{SI_j} Soft iron coefficients of matrix (j=1...9).

c_j Burckhardt tyre model coefficients (j=1,...,5).

d Distance from center of gravity to right wheels.

$\bar{\mathbf{d}}$ Distance vector from the CG to the IMU location.

e_i Chassis quarter height displacement (i=1,...,4).

$\mathbf{e}_k$ State error between measure and estimate.

f_j Generic non-linear equation of index j.

$\bar{\mathbf{g}}$ Gravity acceleration vector.

h_i Center of wheel i height (i=1,...,4).

m Vehicle plus driver mass.

m_{ch} Vehicle chassis mass.

m_{u_i} Vehicle unsprung mass i.

m_x, m_y, m_z Magnetic field components.

$\mathbf{m}$ Magnetic vector.

$\mathbf{m}_C$ Magnetic vector calibrated.

$\mathbf{m}_r$ Magnetometer reading vector.

p_x^{cg}, p_y^{cg} CG position coordinates.

$\bar{\mathbf{p}}_k$ Discrete position vector.

$\bar{\mathbf{p}}_r$ Position reading vector from sensor.

r_a, r_b Ellipse axis.

$\bar{\mathbf{r}}_i$ Position vector of wheel i to the CG.

s_l, s_s, s_r Wheel slip longitudinal, lateral and resulting.

tr Front a rear axle size (track).

$\mathbf{u}_k$ Discrete input vector.

v_R Rotational equivalent wheel velocity.

v_W Wheel linear velocity.

v_{cg} Velocity module at center of gravity.

v_x, v_y Velocity components.

$\mathbf{v}_k$ Discrete observation noise vector.

$\mathbf{v}_{pk}$ Discrete position observation noise.

$\bar{\mathbf{v}}_{\psi k}$ Discrete filter yaw noise.

$\bar{\mathbf{v}}$ Velocity vector of the vehicle.

$\bar{\mathbf{v}}_i$ Velocity vector of wheel i.

$\bar{\mathbf{v}}_k$ Discrete velocity vector.

w_δ Noise of steering encoder reading.

$w_{\dot{\phi}_r}, w_{\dot{\theta}_r}, w_{\dot{\psi}_r}$ Noise of gyroscope reading components.

w_{ψ_r} Noise of yaw reading.

$w_{a_x r}, w_{a_y r}, w_{a_z r}$ Noise of accelerometer reading components.

w_{x_r}, w_{y_r} Noise of position reading components.

$\mathbf{w}_{\dot{\psi}}$ Continuous yaw rate state noise.

$\mathbf{w}_{\dot{\psi}k}$ Discrete filter yaw rate state noise.

$\mathbf{w}_{b_{\dot{\psi}}k}$ Discrete filter bias yaw rate state noise.

$\mathbf{w}_k$ Discrete filter state noise vector.

$\mathbf{w}_{v_y}$ Lateral velocity noise state.

$\bar{\mathbf{w}}_{a_r k}$ Discrete filter acceleration reading noise.

$\bar{\mathbf{w}}_{b_a k}$ Discrete filter acceleration bias noise.

$\bar{\mathbf{w}}_{pk}, \bar{\mathbf{w}}_{ak}$ Discrete filter position and acceleration noise.

x_i, y_i Position components of wheel i.

x_p, y_p, z_p ENU position coordinates.

x_r, y_r Position reading components.

x_{rec} Longitudinal distance for the receiver location.

$\mathbf{x}_k$ Discrete state vector.

$\mathbf{z}_k$ Discrete observation vector.

Subscripts

0 Initial value.

i Quarter suspension index.

k Discrete time index.

x, y, z Cartesian components.

Superscripts

$\{-1\}$ Inverse.

$\{-\}$ Indication of variable computed in previous instant.

$B, 1, E$ Body and Earth reference frames.

$F, 1, R$ Front and rear.

FL, FR, RL, RR Front left, front right, rear left and rear right.

T Transpose.

Others

$\hat{x}$ Estimated value of x.

$\bar{x}$ Vector of x values.

$\dot{x}$, $\ddot{x}$ First and second derivative of x to time (velocity and acceleration).

Glossary

ACF	Attitude Complementary Filter.
CG	Centre of gravity.
CPU	Central Processing Unit.
CTR	Current Transition Ratio.
ECEF	Earth-Centered, Earth-fixed.
ENU	East North Up.
ESC	Electronic Speed Controller.
GPIO	General Purpose Input/Output.
GPS	Global Positioning System.
IMU	Inertial Measurement Unit.
INS	Inertial navigation system.
I^2C	Inter-Integrated Circuit.
LED	Light Emitting Diode.
LTI	Linear Time Invariant (system).
PCF	Position Complementary Filter.
PWM	Pulse-Width Modulation.
RC	Radio Controlled.
TCP	Transmission Control Protocol.
UAS	Uniformly Asymptotically Stable.
UAV	Unmanned aerial vehicle.
UDP	User Datagram Protocol.
UKF	Unscented Kalman Filter.
USB	Universal Serial Bus.
VLE	Vehicle Linear Estimator.
VNLE	Vehicle Non-Linear Estimator.
dof	Degrees of freedom.
rpm	Rotations per minute.

Chapter 1

Introduction

1.1 Motivation

Autonomous driving was been an area of active research in the last decades. For instance, in the 60's, there were already projects undergoing to create highways specific for autonomous driving, with promised of no hands-driving, self-parking or increased safety [1].

Over the past years, many mobile robotic competitions have been created, that require the combination of theoretical results and engineering development, and have been attracting the public audience as well as big companies [2]. Some competitions like the Robocup serve more as an academic purpose, others like MBZIRC and Darpa Challenges are used to make a step forward in technology evolution to be applied in day-by-day applications.

In recent years, several companies like Mercedes, Google and Tesla have been transposing these technologies to commercial road vehicles. At the time no public available vehicle uses a full driverless automation or level 5. The automated driving levels are defined by the SAE International Standard J3016 [3]. Where the level zero is the standard for no automation, level 5 is the full automation where the vehicle takes care of all aspects of the dynamic driving task, in every road and environmental conditions. The current technology is between the level two and three. Where level two stands for partial automation, where the vehicle uses one or more driver assistance system for steering, acceleration and braking, but is expected that the driver takes care of every remaining tasks. The level three stands for conditional automation, where the vehicle uses the drivers assistance systems to drive itself but expects that the driver will respond to a requested intervention.

Connecting these two worlds, of mobile competitions and autonomous road vehicles, is Formula Student with its new driverless competition. Formula Student is the biggest educational engineering competition that challenges university students to build and race single seated racing cars. Founded in 1980 in the U.S.A., has been in constant evolution from adapted motorcycle engines and tubular steel frames, to a massive use of composite materials and independent four-wheel drive with electric motors. These advances also lead to an evolution of control strategies implemented on the cars, namely traction control, stability control, lunch control and torque-vectoring (electronic differential) to two and

four wheels.

Most recently, Formula Student opened a new parallel competition for driverless cars, where the vehicles, tracks and all the competitions characteristics remain the same with the single difference of no human on board. This requires another big improvement in control systems applied in the cars, but for that more sensors and more information on the car state are necessary .

Control strategies like vehicle stability control and torque-vectoring rely on sideslip observers to assure that the vehicle remains in a stable condition [4]. The sideslip angle of a car is defined as the angle between the velocity vector and the heading of the vehicle. Even though many advanced control algorithms may benefit of a sideslip angle observation to improve its performance or reliability, this information is even more necessary in autonomous vehicles. When the car has a driver, this one can do small adjusts to compensate over-steering and under-steering by means of steering wheel or reliving the accelerator. However, a driverless car must have some information about the sideslip to be able to make the same kind of adjustments.

1.2 State of the Art

The need for a sideslip angle value is not a new problem, and several approaches have already been explored. In order to acquire this angle two types of sensors are available.

The first is an optical sensor or optical flow, that computes the velocity components by the apparent motion between two images. One of the most commonly used in road vehicles is the Correvit® family sensors as used by [5]. This kind of sensors, since they work with image acquisition require some considerable computational power that comes in a form off a logging unit associated with the sensor. Also, these sensors require regular calibrations to maintain their performance. In short, this type of sensors is not small, requires constant calibrations and has acquisitions prices in the order of several thousands. But on the other hand, has high accurate results.

The second one is a differential global positioning system (DGPS). This acquisition system consists in using two receiver antennas on the vehicle, and a dedicated system to combine the two acquisition sources and to make reliable measurements. These sensors are also expensive, and are susceptible to the same error sources as the normal GPS, like bad acquisition data in urban environments. Also, the accuracy of this system is dependent of the distance between receivers. Long vehicles like trucks and boats have very good results, while small vehicles like urban cars have worse.

Since these dedicated sensors are expensive, several works try to estimate this angle by other means. The most common is by using inertial sensors and/or GPS as done by [6] [7] [8]. These works usually use a planar model approximation as the system model, like [5] [7] [9] [10], some of them also explore the influence of the vehicle roll [8] [11]. Most of them use common linear or non-linear Kalman filters, KF or EKF, for the sensors integration, with some exceptions like [5] that explores the Unscented Kalman Filter (UKF) instead of the EKF. Besides these algorithms some other authors try different approaches like [12] that uses fuzzy models to minimize some erroneous estimates, and [13]

that makes use of neural networks. Some other works tried a different approach than the INS/GPS systems, by measuring the tyre forces using wheel-force transducers like [5] [9] [11]. In general, all these works are made for common road vehicles or for small electric cars.

In order to combine different sensors, sensor fusion algorithms are used. The INS/GPS fusion has been widely explored along the years [14] [15], and specifically in different types of vehicles like unmanned aerial vehicles (UAV) [16] and in land vehicles [17]. Another strand of sensor fusion is using complementary filters, where it is used the low-frequencies of a sensor and combined with the high-frequencies of another. This is an algorithm explored by [18] and [19].

1.3 Thesis Objectives and Contributions

The primary objective of this work is to provide an estimator architecture that can do the same as the dedicated sideslip sensors (optical flow and DGPS) within acceptable error, and that can be viable enough to be implemented an be an integral part of a Formula Student racing prototype, something that is not possible with the dedicated sideslip sensors due to size.

Additionally, dedicated sensors need regular calibrations or long warmups times. The proposed architecture will be a self-calibrating system, capable of adapting to changes and drifts in the sensor readings.

This work also aims to be a starting point for an estimator adaptable to any four wheel road vehicle, capable of returning filtered real time information of common quantities as well as, parameters that previously could not be measured and that remained unknown like lateral tyre forces.

The vehicle model presented, is expected to be a platform for testing filters and estimators saving time in implementations and road tests. Also, due to its modules is expected to be upgradable with more complex modules.

In [20] was developed the communication of a remote controller to a computer by USB, using Simulink, which send it the values through wireless. In this thesis, the this work is continued by developing the C program that receives the values through wireless and encodes it to PWM signals in order to control the motor and the servo of the RC vehicle.

Along this work, a generic library in C code was developed for Kalman Filter, used in Section 5, that is expected to facilitate the integration of filters and estimators in the RC vehicle micro-controllers. Being C code a widely used language, from micro-controllers to personal computers it can easily be transposed to different systems.

1.4 Thesis Outline

Following this introductory section, Chapter 2 introduces the mathematical formulation for the vehicle model to be used in the simulations and to adjust and test the estimators. The several sub models are explained in detail and the assumptions used in each one, as well as the limitations.

In Chapter 3 the estimator architecture is introduced. In a first part the Kalman Filter and the Extended Kalman Filter are explained. They are the starting point for the proposed estimators. Then the estimators used are presented, beginning with the complementary kinematic filters for position and attitude, and followed by the vehicle estimators with a linear and a non-linear models already presented in the previous chapter.

Chapter 4 presents the simulations results. The vehicle model exposed in the Chapter 2 is used to recreate the behaviour of a real vehicle, and the estimator architecture proposed in Chapter 3 is tested and compared with the true values.

In Chapter 5, the estimator architecture is implemented in a micro-controller on a RC car inside a laboratory. In this test the proposed estimator is receiving real time data from the onboard sensors, and computing the sideslip angle, also in real time. The results are compared with the values acquired by a high accuracy position acquisition system.

In Chapter 6 data from a real Formula Student prototype was acquired and is used to feed the estimators. This test is made in offline, and the data crossed with the results of a differential GPS.

Chapter 7 will present some conclusions and some suggestions for future work to help improve results and implementation.

Chapter 2

Vehicle Model

This chapter presents a formulation for a dynamic car to test the estimators. A car is a complex system that can be decomposed in several sub-models. In Fig.2.1 the subsystems used are shown and how they are interconnect.

Under normal conditions, the forces applied in a car come from two major interactions, tyres and aerodynamic devices. These forces are supplied to the planar model which is responsible for all the horizontal movement of the vehicle. The accelerations ($\dot{v}_x$ and $\dot{v}_y$) created by this planar movement generate load transfers that are supplied to the vertical model, which combined with the vertical component of the aerodynamic, widely known as downforce (F_L), creates the load in each tyre (F_{z_i}). Also a track input is used to simulate road inclination or road banking by an height input at each wheel (G_i). The tyre model uses this vertical force as well as the velocity components and the steering angle (δ) to generate the tyre lateral forces (F_{y_i}) supplied to the planar model.

The driver only has control of the steering angle and the acceleration and brake pedals. These last two are combined and simplified as a longitudinal force (F_x) which is considered a direct input on the vehicle to which is subtracted the aerodynamic longitudinal force known as drag (F_D).

The last model is the sensors. All the values and states described before are not accessible in a real vehicle in an absolute form as in a model. Thus, this block recreates the readings from a GPS (Global

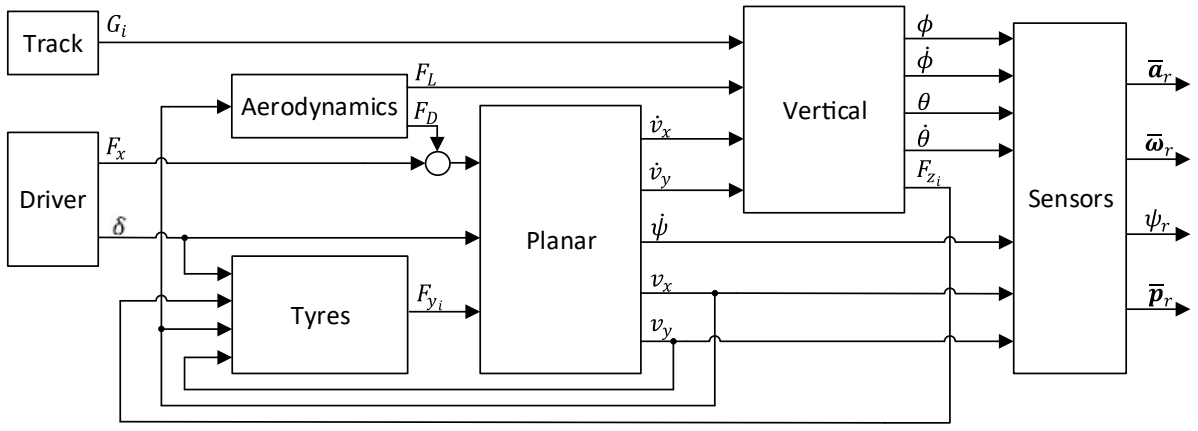

Figure 2.1: Car model diagram and interconnection between sub-models

Positioning System) for the position ($\bar{\mathbf{p}}_r$), the accelerations ($\bar{\mathbf{a}}_r$) from an accelerometer, the angular velocities from a gyroscope ($\bar{\omega}_r$) and the heading (ψ_r) from a digital compass. This is done using the longitudinal (v_x) and lateral (v_y) velocities and yaw rate ($\dot{\psi}$) from the planar model as well as the roll (ϕ) and pitch (θ) angles and the roll rate ($\dot{\phi}$) and pitch rate ($\dot{\theta}$) given by the vertical model.

Besides the sub-models presented above, this chapter also contains two additional linear models one for the tyres, and the other for the planar movement that are used ahead in the estimators.

2.1 Tyres

Recreating a tyre behaviour is a well-known problem and several models have been presented like Lumped Models [21], Brush Model [22], Tread Simulation Model [22], TMeazy [23], Burckhardt [24] and the most known and accepted, the Magic Formula of Pacejka [25]. Most of them use equations that should be adjusted to real tyre data.

In Fig.2.2 a Cartesian coordinate system is attached to a tyre, where the three forces and three moments that act on a tyre are depicted. The longitudinal force (F_x) that appears on driving wheels while accelerating, and on all wheels while braking. The lateral force (F_y), that makes the car turn, it only appears when the side slip angle of the wheel $\alpha_i \neq 0$. The wheel load (F_z), is the weight of the car on that wheel. Roll moment (M_ϕ) contradicts the camber angle (γ) and can also generate lateral forces. Pitch moment (M_θ), also called rolling resistance, counteracts the wheel rotation. Yaw moment (M_ψ) also called self aligning moment, opposes the steering angle.

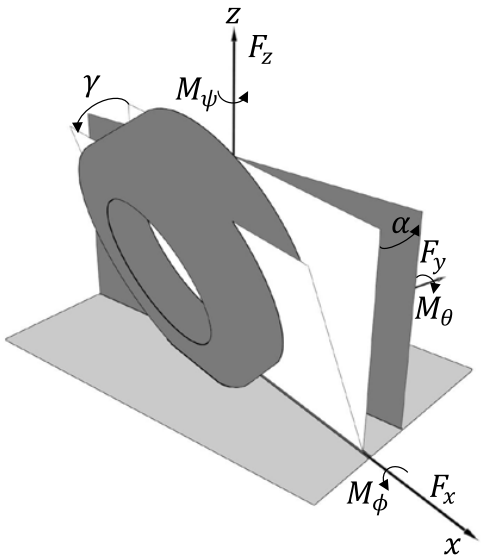

Figure 2.2: Tyre coordinate system [26].

Some of the models already mentioned before are very precise at fitting experimental data, but consequently, they are extremely complex and computationally heavy with several coefficients that need to be calculated. Since this work's main objective is not to simulate all the non-linearities of a tyre, and all the efforts that it is subjected, the Burckhardt model was chosen, mostly due to its simplicity, but also the ability to recreate the most significant characteristics of a tyre.

2.1.1 Burckhardt

The Burckhardt model starts by the definition of longitudinal and lateral slip of a wheel in driving and braking conditions (Tab.2.1), ensuring that each one is limited between -1 and 1. Where are used the definitions of wheel linear velocity (v_W), and rotational equivalent wheel velocity (v_R).

	Braking	**Driving**
	$v_R \cos\alpha \leq v_W$	$v_R \cos\alpha > v_W$
Longitudinal Slip (s_l)	$\frac{v_R \cos\alpha - v_W}{v_W}$	$\frac{v_R \cos\alpha - v_W}{v_R \cos\alpha}$
Lateral Slip (s_s)	$\frac{v_R \sin\alpha}{v_W}$	$\tan\alpha$

Table 2.1: Tyre longitudinal and side slip definitions [24]

Instead of splitting the circle of Forces (or Kamm Circle) in longitudinal and lateral, this model assumes equal conditions on both directions and works with a resultant slip given by (2.1) in order to calculate a resulting friction coefficient (μ_r).

$$s_r = \sqrt{s_l^2 + s_s^2} \tag{2.1}$$

$$\mu_r(s_r, F_z, v_{cg}) = (c_1(1 - e^{-c_2 s_r}) - c_3 s_r) e^{-c_4 s_r v_{cg}} (1 - c_5 F_z^2) \tag{2.2}$$

Typical values for the coefficients of (2.2) are proposed by the Burckhardt model, where the three first are given by Tab.2.2 for common road cars in different road conditions.

	c1	c2	c3
Asphalt, dry	1.2801	23.99	0.52
Asphalt, wet	0.857	33.822	0.347
Concrete, dry	1.1973	25.168	0.5373
Cobblestones, dry	1.3713	6.4565	0.6691
Cobblestones, wet	0.4004	33.7080	0.1204
Snow	0.1946	94.129	0.0646
Ice	0.05	306.39	0

Table 2.2: Parameter sets for friction coefficient (Burckhardt [24])

The fourth (c_4) is the velocity adjust parameter that is said to be between 0.002s/m and 0.004s/m. The c_5 is introduced as close to 0.00015(1/kN)^2. The resulting force generate by the tyre is then given by the Coulomb friction equation (2.3).

$$F_r = \mu_r(s_r, F_z, v_{cg}) F_z \tag{2.3}$$

2.1.2 Tyre Model adaptation

Even though the Burckhardt model has values for several ground conditions, they are for regular tyres, where the tyres on a formula student prototype are racing slick tyres with different compounds than the regulars, that are made for better traction at the cost of higher tyre wear. In order to have a more realistic result, real tyre data provided by the "Formula SAE Tire Test Consortium (FSAE TTC)" was used to fit results with equation (2.2), for the same tyres used in the real prototype.

The data provided by the FSAE TTC [27], includes several channels like F_x, F_y, F_z, M_ϕ and M_ψ already mentioned before, temperature at 5 locations, wheel and road speed, pressure, loaded radius, inclination angle, slip ratio (s_r) and slip angle. As said before the Burckhardt model is a simpler model and does not consider all these values, besides in this work the longitudinal force is assumed as a direct input and so the slip ratio does not have any influence. With this, only the lateral force (F_y), the slip angle (α), and the normal force (F_z) are considered from the real data for the fit. The remaining variables available from the raw data were kept at fixed values as close to a neutral value as possible, to minimize the influence in the fit results.

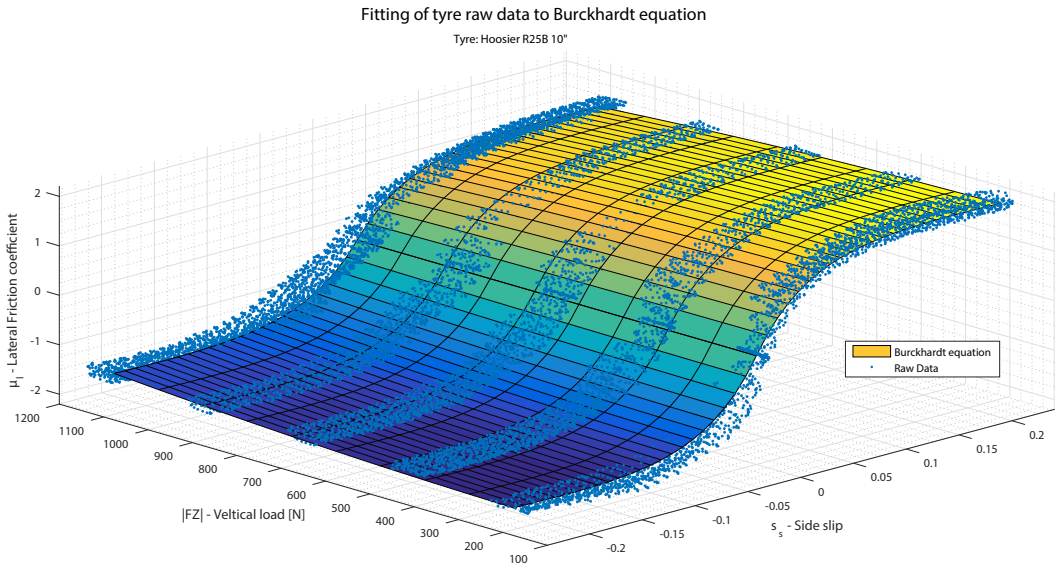

Figure 2.3: Raw tyre data fit to Burckhardt method. With a $R^2 = 0.944$

In this approximation, the influence of the velocity (v_{cg}) in the traction coefficient was neglected due to lack of data, and therefore c_4 is assumed as null. The fit presented in the Fig.2.3, relied in 15633 data points, and resulted in the coefficients presented in equation (2.4). It should be noted that equation (2.2) only presents the data in the first quadrant, and for the sake of data presentation and similarity with common side slip graphs, the third quadrant was added using the mirrored equation. No alteration was made to the raw data.

$$F_y(s_r, F_z) = (2.013 \times (1 - e^{-20.294 s_r}) - 0.966 s_r)(1 - 1.837 \times 10^{-7} F_z^2) \times F_z \tag{2.4}$$

$$s_r = |\tan \alpha| \tag{2.5}$$

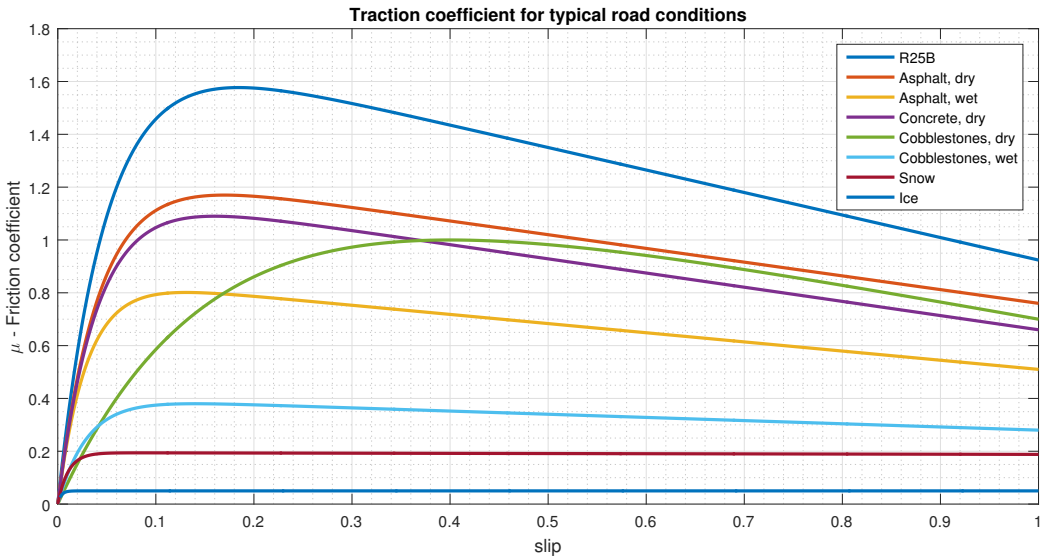

Figure 2.4: Comparison between several types of common road surfaces plus the Hossier R25B in asphalt

It's then possible to resume these models for the required application to (2.4) which results in the combination of (2.2), (2.3) and the results of the fit presented in Fig.2.3. Where the s_r is given by (2.5) the combination of Tab.2.1 with equation (2.1), using the assumption of only pure cornering while driving, and neglecting the influence of longitudinal slip.

In Fig.2.4 is possible to see a comparison between the several types of road an tyres interaction as seen in Tab.2.2, as well as the data from the FSAE TTC for the R25B tyre.

It is well known that this model is far from reality, due to a simplistic method (Burckhardt), ignoring the longitudinal slip for the calculation of the available lateral force, and several others approximation mentioned before. But despite that, this model held a few of the most important characteristics of the tyre behaviour, which are the saturation of the lateral force at some slip angle value and even a reduction of this force, a peak value for the force, and a major influence of the tyre's vertical load.

2.1.3 Linear Tyre Model

A regular approximation used in linear models is the substitution of a tyre model for a cornering stiffness constant (C_α) [26][28]. This value is defined as the initial slope of a function $F_y = f(\alpha)$. Several problems arise from this approximation. The range of angles from where the slope is calculated is not fix, and as can be seen in Fig.2.5 choosing a wider range of values (larger angles) result in a loss of information when using low values of α, which results in a force considerably lower than the actual one. On the other hand, choosing a smaller range of α does not give enough information for more aggressive driving conditions. Another problem is the independence of the normal force that does not influence with the lateral force. And the major problem is the fact that the force never saturates. If the value of α used exceeds the assumed ranged, then the calculated lateral force will have no meaning since it will surpasses the maximum peak force of the tyre by far. As a simple example, using the second linear

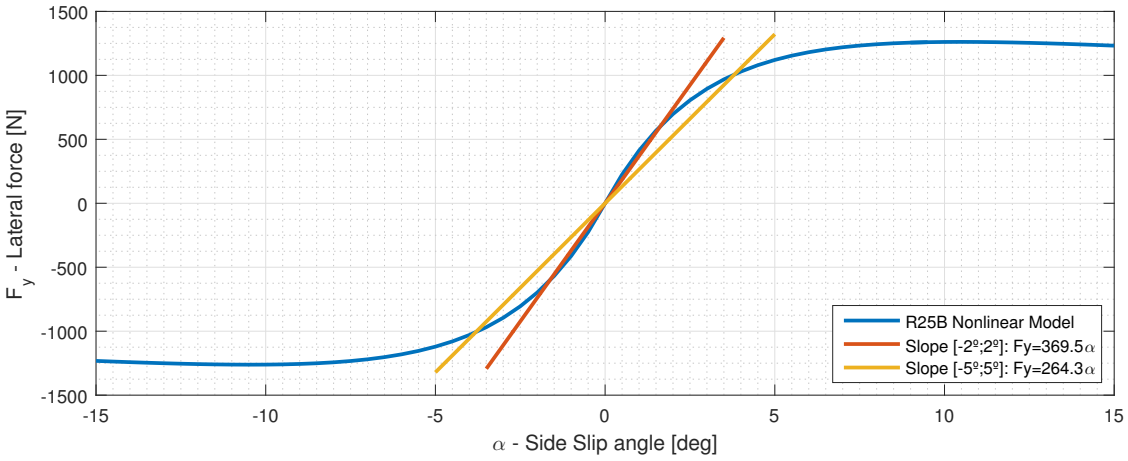

Figure 2.5: Tyre Model: Linear (cornering stiffness) vs NonLinear (Burckhardt), for a vertical load of Fz=900N.

equation of Fig.2.5, $F_y = 264.3\alpha$, if the slip angle surpass the defined range even if only by 0.5º, the lateral force will be 15% more than the maximum peak force the tyre can produce.

In conclusion, this linear approximation is only acceptable for very low values of the tyre's side slip.

2.2 Vertical Model

To model the vertical dynamics of the car as well as the rotations of roll and pitch, a 7-dof simplified vertical model is presented in Fig.2.6. In the literature is quite common to encounter suspension models as quarters of a car simplified with two mass-spring-damper systems in series [26][28]. The proposed model is a combination of the four quarters of the car to also consider the interactions between wheels,

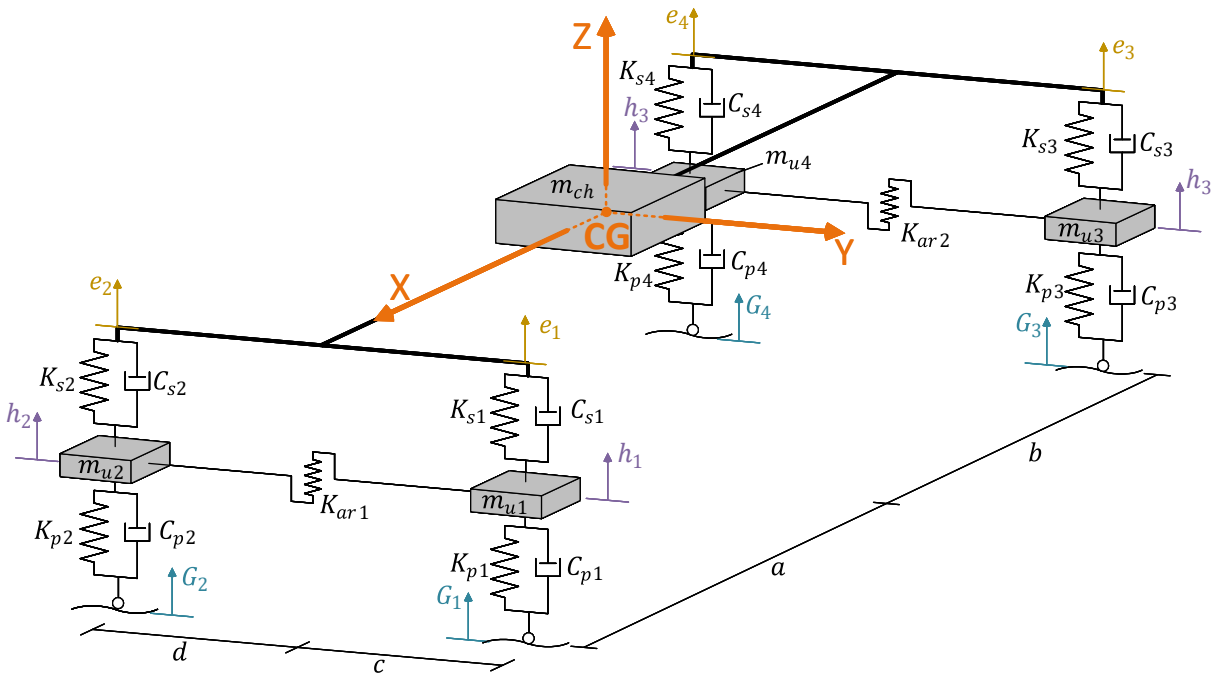

Figure 2.6: Vertical Model

something that a quarter suspension models does not consider.

The degrees of freedom (dof) of this model are, the roll (ϕ), the pitch (θ), vertical position (Z), and the height of each centre wheel (h_i), where i=1,2,3,4 designates the suspension quarters. The model components are a chassis of mass (m_{ch}) and inertia to roll and pitch respectively I_ϕ and I_θ, four unsprung masses (m_{ui}) representing the wheel, tyre, brake, upright and all the remaining components that are not suspended by the shock absorbers. Each suspension quarter (i) is defined as an equivalent linear spring damper parallel system, where K_{si} is the spring stiffness and C_si the damping coefficient. The tyres are also represented as a spring damper system where K_{pi} is the tyre stiffness coefficient and C_{pi} the damping coefficient. For each axle an equivalent spring (K_{arj}, j=1,2) connects the unsprung masses to represent the anti-roll bar.

The inputs of this model are, moments applied to roll (M_ϕ) and pitch (M_θ), vertical load (F_z) and ground height applied to each wheel (G_i, i=1,2,3,4). To keep this model linear, a small angle approximation is performed, where the displacements of the top suspension quarters are:

$$\begin{aligned} e_1 &= Z + c\phi - a\theta \quad , \qquad e_2 = Z - d\phi - a\theta \\ e_3 &= Z + c\phi + b\theta \quad , \qquad e_4 = Z - d\phi + b\theta \end{aligned} \tag{2.6}$$

For simplification of the equations is defined a $\Delta_{si} = e_i - h_i$ and $\dot{\Delta}_{si} = \dot{e}_i - \dot{h}_i$ for i=1,...,4. A balance of forces (2.7)-(2.13) is performed for each degree of freedom.

$$m_{ch}\ddot{Z} + \sum_{i=1}^{4} K_{si}\Delta_{si} + \sum_{i=1}^{4} C_{si}\dot{\Delta}_{si} = F_z \tag{2.7}$$

$$I_\theta\ddot{\theta} + a\left(\sum_{i=1}^{2} K_{si}\Delta_{si} + \sum_{i=1}^{2} C_{si}\dot{\Delta}_{si}\right) - b\left(\sum_{i=3}^{4} K_{si}\Delta_{si} + \sum_{i=3}^{4} C_{si}\dot{\Delta}_{si}\right) = M_\theta \tag{2.8}$$

$$I_\phi\ddot{\phi} - c\left(\sum_{i=1,3} K_{si}\Delta_{si} + \sum_{i=1,3} C_{si}\dot{\Delta}_{si}\right) + d\left(\sum_{i=2,4} K_{si}\Delta_{si} + \sum_{i=2,4} C_{si}\dot{\Delta}_{si}\right) = M_\phi \tag{2.9}$$

$$m_{u1}\ddot{h}_1 - K_{s1}\Delta_{s1} - C_{s1}\dot{\Delta}_{s1} + K_{p1}(h_1 - G_1) + C_{p1}(\dot{h}_1 - \dot{G}_1) + K_{ar1}(h_1 - h_2) = 0 \tag{2.10}$$

$$m_{u2}\ddot{h}_2 - K_{s2}\Delta_{s2} - C_{s2}\dot{\Delta}_{s2} + K_{p2}(h_2 - G_2) + C_{p2}(\dot{h}_2 - \dot{G}_2) + K_{ar1}(h_2 - h_1) = 0 \tag{2.11}$$

$$m_{u3}\ddot{h}_3 - K_{s3}\Delta_{s3} - C_{s3}\dot{\Delta}_{s3} + K_{p3}(h_3 - G_3) + C_{p3}(\dot{h}_3 - \dot{G}_3) + K_{ar2}(h_3 - h_4) = 0 \tag{2.12}$$

$$m_{u4}\ddot{h}_4 - K_{s4}\Delta_{s4} - C_{s4}\dot{\Delta}_{s4} + K_{p4}(h_4 - G_4) + C_{p4}(\dot{h}_4 - \dot{G}_4) + K_{ar2}(h_4 - h_3) = 0 \tag{2.13}$$

The expansion of these equations is performed in Appendix A, and the system of equations is used as a state space system. The state variables are [Z, $\dot{Z}$, θ, $\dot{\theta}$, ϕ, $\dot{\phi}$, h_1, $\dot{h}_1$, h_2, $\dot{h}_2$, h_3, $\dot{h}_3$, h_4, $\dot{h}_4$]$^\mathsf{T}$ and the inputs of the system [F_z, M_θ, M_ϕ, G_1, G_2, G_3, G_4]. The necessary roll (ϕ) and pitch (θ) angles are outputs of the system, while the tyre vertical load is computed by (2.14) assuming the tyre as a linear spring damper [29].

$$F_{zi} = K_{pi}(h_i - G_i) + C_{pi}(\dot{h}_i - \dot{G}_i), \quad i = 1, 2, 3, 4 \tag{2.14}$$

2.3 Planar Car Model: Non-Linear

As the name suggest, the planar car model implies that the vehicle only moves in two directions. Assuming the car as a rigid body, the dynamics can be expressed by the Newton-Euler equations of motion. Considering a reference frame attached to the centre of gravity (CG), results in the following balance.

$$F_x = m\dot{v}_x - m\dot{\psi}v_y \tag{2.15a}$$

$$F_y = m\dot{v}_y + m\dot{\psi}v_x \tag{2.15b}$$

$$M_\psi = \ddot{\psi}I_\psi \tag{2.15c}$$

Where $\dot{\psi}$ is the angular velocity around Z-axis, that is, the yaw rate of the car, v_x and v_y are the longitudinal and transversal velocities, respectively, F_x and F_y the longitudinal and lateral forces, m is the mass of the vehicle plus driver, and I_ψ is the inertia around the Z-axis. Without the aerodynamic influence, the forces acting on the car are expressed in Fig.2.7, all of them depending on the tyres. The tyres indexation used is front left (FL), front right (FR), rear left (RL) and rear right (RR). It's assumed the car is front steer only, and the steering angle δ is assumed equal on the two front wheels. The distance between the CG and the front axle is a, and from the CG to the rear axle is b, also the track (tr) front and rear is assumed equal.

$$\dot{v}_x = v_y\dot{\psi} - \frac{1}{m}[F_y^F \sin\delta - F_x^F \cos\delta - F_x^R] \tag{2.16a}$$

$$\dot{v}_y = -v_x\dot{\psi} + \frac{1}{m}[F_y^F \cos\delta + F_y^R + F_x^F \sin\delta] \tag{2.16b}$$

$$\ddot{\psi} = \frac{1}{I_\psi}a[F_y^F \cos\delta + F_x^F \sin\delta] - \frac{1}{I_\psi}bF_y^R \tag{2.16c}$$

Combining (2.15) with the forces applied on the car (Fig.2.7), results the general balance (2.16),

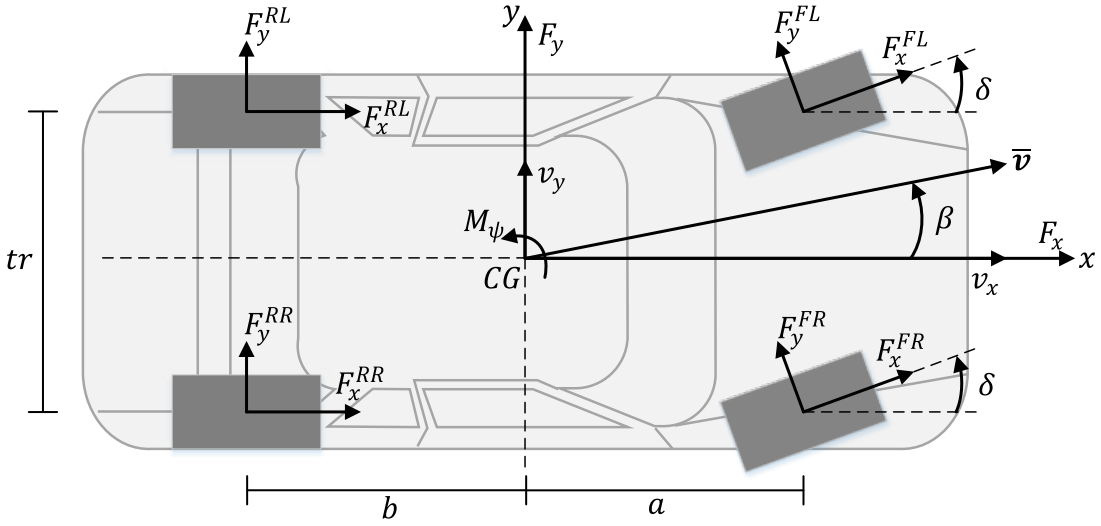

Figure 2.7: Forces applied on the vehicle

where any kind of moment created by a mechanical or electrical differential is neglected, and the simplifications of (2.17) were applied, using F for front and R for rear.

$$\begin{aligned} F_x^F &= F_x^{FL} + F_x^{FR} \quad , \qquad F_x^R = F_x^{RL} + F_x^{RR} \\ F_y^F &= F_y^{FL} + F_y^{FR} \quad , \qquad F_y^R = F_y^{RL} + F_y^{RR} \end{aligned} \tag{2.17}$$

The longitudinal forces are used as a direct input and since the car is assumed rear driven only, $F_x^F = 0$. The lateral forces depend on the tyre model, and both the linear and the non-linear (2.4) models rely on the tyre slip angle.

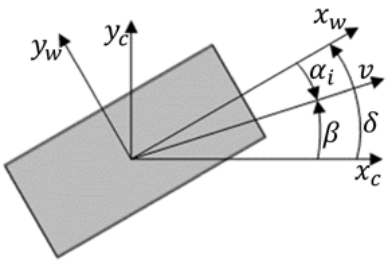

Figure 2.8: Wheel vectors. The w and c suggest the wheel and car reference frames respectively, and v is the velocity vector

The sideslip of car (β) is defined as the angle between the heading of the car and the velocity vector, and as can be seen in Fig.2.7, this angle is given by (2.18) where the velocities are given in the vehicle reference frame. The slip angle α_i of a tyre, as seen in Fig.2.8 is given by (2.19), where β_i is the projected sideslip of the car on the wheel i.

$$\beta = \text{tg}^{-1}\left(\frac{v_y}{v_x}\right) \tag{2.18}$$

$$\alpha_i = \beta_i - \delta_i \tag{2.19}$$

Assuming that the wheel i is at a distance $\bar{\mathbf{r}}_i = (x_i, y_i)$ of the CG, and ${}^B\bar{\mathbf{v}}$ is the vector with the velocity components at the centre of gravity, then the velocity components at the wheel ${}^B\bar{\mathbf{v}}_i$ can be achieved by (2.20).

$${}^B\bar{\mathbf{v}}_i = {}^B\,\bar{\mathbf{v}} + \bar{\omega} \times \bar{\mathbf{r}}_i = \begin{bmatrix} v_x \\ v_y \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} x_i \\ y_i \\ 0 \end{bmatrix} = \begin{bmatrix} v_x - y_i\dot{\psi} \\ v_y + x_i\dot{\psi} \\ 0 \end{bmatrix} \tag{2.20}$$

Merging the result of equation (2.20) with (2.18) and (2.19), results in the general equation (2.21) for the wheel slip angle. It should be noted that for rear wheels, δ_i=0 due to assumption of front steer only.

$$\alpha_i = \text{tg}^{-1}\left(\frac{v_y + x_i\dot{\psi}}{v_x - y_i\dot{\psi}}\right) - \delta_i \tag{2.21}$$

2.4 Planar Car Model: Linear

The car model presented above is highly non-linear, which is problematic for estimation, control and requires much more computational power. A linear approximation has been widely used in literature [6][11][26] which relies on the model presented before (2.16). This linear model has three major assumptions, constant longitudinal velocity, small angle approximation, and is based on a bicycle model.

The bicycle model assumes there is no y dimension which combined with the small angle approximation results in a new equation for the side slip angle for each wheel (2.22). Besides that, for this model is used the linear tyre model (2.23) based on the cornering stiffness definition. Note that the negative sign is due to the referential in use.

$$\alpha_i = \text{tg}^{-1}\left(\frac{v_y + x_i\dot{\psi}}{v_x - y_i\dot{\psi}}\right) - \delta_i = \frac{v_y + x_i\dot{\psi}}{v_x} - \delta_i \tag{2.22}$$

$$F_{y_i} = -C_\alpha \alpha_i \tag{2.23}$$

Recalling the model (2.16), using the the small angle approximations $\sin 0 \approx 0$, $\cos 0 \approx 1$, and equation (2.23), the new balance for the bicycle model is defined as (2.24). Assuming that all tyres are equal, and that left and right normal loads are the same, then C_α^{FL}+C_α^{FL}=C_{α_f} and analogously the same for the rear wheels. Additionally, since there's no y dimension in the bicycle model, then left and right α_i are equal.

$$\dot{v}_x = v_y\dot{\psi} + \frac{1}{m}[F_x^F + F_x^R] \tag{2.24a}$$

$$\dot{v}_y = -v_x\dot{\psi} - \frac{1}{m}[C_{\alpha f}\alpha_f + C_{\alpha r}\alpha_r] \tag{2.24b}$$

$$\ddot{\psi} = -\frac{1}{I_\psi}aC_{\alpha f}\alpha_f + \frac{1}{I_\psi}bC_{\alpha r}\alpha_r \tag{2.24c}$$

Due to the assumption of constant longitudinal velocity, the first equation of (2.24) falls, and the remaining are combined with (2.22), resulting in:

$$\dot{v}_y = -v_x\dot{\psi} - \frac{C_{\alpha f} + C_{\alpha r}}{mv_x}v_y - \frac{aC_{\alpha f} + bC_{\alpha r}}{mv_x}\dot{\psi} + \frac{C_{\alpha f}}{m}\delta \tag{2.25a}$$

$$\ddot{\psi} = -\frac{aC_{\alpha f} + bC_{\alpha r}}{I_\psi v_x}v_y + \frac{-a^2C_{\alpha f} + b^2C_{\alpha r}}{I_\psi v_x}\dot{\psi} + \frac{aC_{\alpha f}}{I_\psi}\delta \tag{2.25b}$$

This system can also be written in a state space form (2.26) which is more useful, where $[v_y, \dot{\psi}]^\mathsf{T}$ are the states and the input is the steering angle δ.

$$\begin{bmatrix}\dot{v}_y \\ \ddot{\psi}\end{bmatrix} = \begin{bmatrix} -\frac{C_{\alpha f} + C_{\alpha r}}{mv_x} & -\frac{aC_{\alpha f} + bC_{\alpha r}}{mv_x} - v_x \\ -\frac{aC_{\alpha f} + bC_{\alpha r}}{I_\psi v_x} & \frac{-a^2C_{\alpha f} + b^2C_{\alpha r}}{I_\psi v_x}\end{bmatrix}\begin{bmatrix}v_y \\ \dot{\psi}\end{bmatrix} + \begin{bmatrix}\frac{C_{\alpha f}}{m} \\ \frac{aC_{\alpha f}}{I_\psi}\end{bmatrix}\delta \tag{2.26}$$

This model as several limitations, being them the validity for only small angles, the tyre model that not only does not represent the minimal dynamics required (like the force saturation) but also does not consider load transfers due to accelerations.

2.5 Aerodynamics

For the aerodynamics is used simple model [28] for the drag and lift (downforce). Moments associated with aerodynamic devices are neglected, being the only forces used the lift (F_L) for the extra load on the tyres at higher velocities and the drag (F_D) for the longitudinal force dissipation. The general equations for these aerodynamic forces are (2.27) for the drag and (2.28) for the lift. Where ρ is the air density, A_x^{proj} and A_z^{proj} are the vehicle area projections in the "x" and "y" directions, C_d and C_l are respectively the drag and lift coefficients of the complete vehicle.

$$C_d = \frac{F_D}{(1/2)\rho A_x^{proj} v_x^2} \tag{2.27}$$

$$C_l = \frac{F_L}{(1/2)\rho A_z^{proj} v_x^2} \tag{2.28}$$

Since the projected areas are always equal for the same vehicle, the lift and drag coefficients are also constant and is assumed the air has always the same density, a simplification (2.29) is made to these equations.

$$\begin{aligned} C_D &= C_d \frac{1}{2} \rho A_x^{proj} \\ C_L &= C_l \frac{1}{2} \rho A_z^{proj} \end{aligned} \tag{2.29}$$

The aerodynamic forces are then given by 2.30 and only depend on the longitudinal velocity of the vehicle to the square.

$$\begin{aligned} F_D &= C_D v_x^2 \\ F_L &= C_L v_x^2 \end{aligned} \tag{2.30}$$

2.6 Sensor Modelling

As stated before, in a real car the values are not available in an absolute and noisy free condition as in the models. To be closer to reality, the available values of the models are reshaped as they are acquired, with offsets and noises. The vehicle is assumed to be equipped with a GPS, a steering encoder and an IMU which consists in a 3-axis accelerometer, a 3-axis gyroscope and a digital compass.

The IMU is attached to the vehicle and rotates with this one, therefore a rotation matrix is defined as (2.31) where R_{imu} is the rotation matrix from vehicle axis to a global axis, ϕ and θ are the roll and pitch angles given by the vertical model.

$$R_{imu} = R_\theta R_\phi = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{2.31}$$

2.6.1 Compass

In an IMU the compass angle is given by a magnetometer, which is corrupted with soft and hard non-linearities [30][31] as well as offsets and noise in all the axes. To simplify, the simulated yaw reading will just be corrupted with Gaussian white noise w_{ψ_r} and given by (2.32) where the yaw angle reading (ψ_r) is given by the time integration of the yaw rate ($\dot{\psi}$) and ψ_0 is the initial heading angle or yaw angle.

$$\psi_r = \psi_0 + \int \dot{\psi} dt + w_{\psi_r} \tag{2.32}$$

2.6.2 Gyroscope

The angular velocities readings ($\bar{\omega}_r$) are given by (2.33) where $\dot{\phi}$ and $\dot{\theta}$ come directly from the vertical model, and $\dot{\psi}$ from the planar model.

$$\bar{\omega}_r = R_{imu} \left(\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} b_{\dot{\phi}} \\ b_{\dot{\theta}} \\ b_{\dot{\psi}} \end{bmatrix} + \begin{bmatrix} w_{\dot{\phi}_r} \\ w_{\dot{\theta}_r} \\ w_{\dot{\psi}_r} \end{bmatrix} \right) \tag{2.33}$$

It is assumed that each rate is corrupted by a constant bias ($b_{\dot{\phi}}$, $b_{\dot{\theta}}$ and $b_{\dot{\psi}}$) and by uncorrelated, zero-mean Gaussian white noise ($w_{\dot{\phi}_r}$, $w_{\dot{\theta}_r}$ and $w_{\dot{\psi}_r}$). Also, the sensor has the rotations of the car, roll and pitch, given by (2.31).

2.6.3 Accelerometer

Before the accelerometer model [32] is presented is important to notice that the accelerations (a_x and a_y) sustained by the vehicle aren't the same as the velocity derivative of the planar model, that is $a_x \neq \dot{v}_x$ and $a_y \neq \dot{v}_y$.

The accelerometer model readings ($\bar{\mathbf{a}}_\mathbf{r}$) are given by (2.34), where $\dfrac{\partial \bar{\mathbf{v}}}{\partial t}$ is the time derivative of the velocity vector $\bar{\mathbf{v}}$=[v_x, v_y, v_z]$^\mathsf{T}$, $\bar{\omega}$=[$\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$]$^\mathsf{T}$ is the angular velocity vector, $\bar{\mathbf{d}}$ is the distance vector from the centre of gravity to the IMU location, and $\bar{\mathbf{g}}$=[0, 0, -9.81]$^\mathsf{T}$ is the gravity acceleration vector.

$$\bar{\mathbf{a}}_\mathbf{r} = R_{imu} \left(\frac{\partial \bar{\mathbf{v}}}{\partial t} + \bar{\omega} \times \bar{\mathbf{v}} + \bar{\omega} \times \left(\bar{\omega} \times \bar{\mathbf{d}} \right) + \bar{\mathbf{g}} + \begin{bmatrix} b_{a_x} \\ b_{a_y} \\ b_{a_z} \end{bmatrix} + \begin{bmatrix} w_{a_x r} \\ w_{a_y r} \\ w_{a_z r} \end{bmatrix} \right) \tag{2.34}$$

The reading are also corrupted with time invariant bias in each axis (b_{a_x}, b_{a_y} and b_{a_z}) and uncorrelated zero-mean, Gaussian white-noise ($w_{a_x r}$, $w_{a_y r}$ and $w_{a_z r}$). The accelerometer readings are also modified by the orientation of the vehicle given by R_{imu}.

2.6.4 Steering encoder

The steering encoder is usually connected to the steering wheel, which is not a linear with the wheel angle due to the suspension geometry of the car where both wheels do not turn the same due to Ackerman geometry [33]. For the sake of simplicity, in this model (2.35) is assumed that both wheels turn the same, and the measured angle is the wheel angle δ with noise associated to the sensor reading w_δ.

$$\delta_r = \delta + w_\delta \tag{2.35}$$

2.6.5 GPS

The GPS in this model works more like a path reconstruction than a common GPS, in the sense that it already provides the processed x and y positions in a global referential than the actual longitude, latitude and altitude coordinates that GPS receivers return.

$$\bar{\mathbf{p}}_r = \begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} \int (v_x \cos\psi - v_y \sin\psi)\, dt + w_{x_r} \\ \int (v_x \sin\psi + v_y \cos\psi)\, dt + w_{y_r} \end{bmatrix} \tag{2.36}$$

Equation (2.36) provides the path reconstruction of the CG by integration of the planar model velocities. It's assumed that both position coordinates have an uncorrelated zero-mean noise (w_{x_r}, w_{y_r}). No bias is added since an offset in location wouldn't have any effect on the estimators used.

Chapter 3

Estimator Architecture

This chapter presents the estimator architecture as well as the algorithms used for the filters and estimators. The main goal of this work is to estimate the sideslip of a car, and for that the proposed algorithm consists in a series of 3 estimators as depicted in Fig.3.1.

First a small overview of the Kalman Filter (KF) and the Extended Kalman Filter (EKF) will be made for their discrete versions, where the equations presented will be the foundation of the estimators used ahead.

The first estimator is an ACF (Attitude Complementary Filter) which uses the readings from the compass and the yaw rate from the gyroscope to provide an attitude estimation and a yaw rate bias estimation to correct yaw rate reading.

A PCF (Position Complementary Filter) follows, that uses the accelerations and the position to give a velocity estimation. Since the position is in a global frame and both accelerations are in the vehicle frame, the attitude estimation from the ACF is used to make the transformation. The PCF also outputs an acceleration bias estimation to correct the accelerometer readings internally.

Both complementary filters are an adaptation of [19] for planar movement and to the present conditions. The attitude filter was reduced from three rotations to only one, and the position filter from three to two axes but the accelerometer bias was introduced.

With perfect readings from the sensors, that is absolute values with no noise, both complementary filters would be enough since the sideslip is defined as $\beta = \mathrm{tg}^{-1}\left(\frac{v_y}{v_x}\right)$, and with the velocity estimates

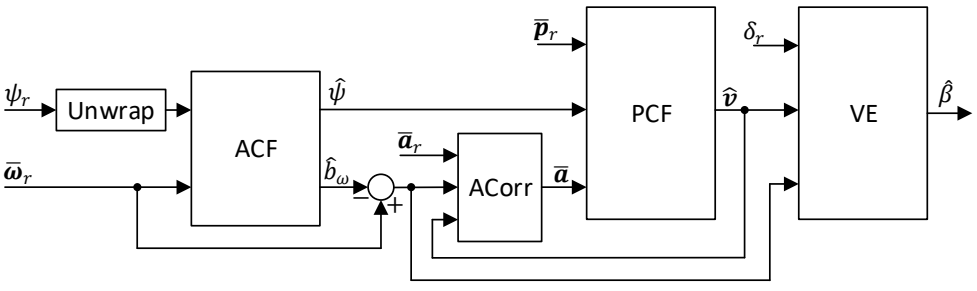

Figure 3.1: Proposed architecture to estimate the sideslip of a vehicle.

it would be a direct math. Due to all the noise associated with the readings, this sideslip computation results in an imperceptible signal. To overcome this situation a third filter is proposed, a vehicle estimator. This filter is based on the planar dynamics of a car. For this vehicle estimator two models are proposed a linear one, and a non-linear one. Further ahead in this work a comparison between both approaches and the direct math using only the complementary filters will be performed, and the pros and cons of each one discussed.

To interconnect these filters two additional blocks are used as seen in Fig.3.1. The first is an unwrap function that converts the yaw angle from a circle range, $[-\pi, \pi]$ to a continuous angle between $[-inf, inf]$ since the original discontinuity is a problem for the ACF. The second one is the "ACorr" block that removes the accelerations induced by the angular velocity on the accelerometer.

3.1 Discrete Kalman Filter

The Kalman Filter and its discrete version are widely explored and documented algorithms [34] [35] [36] [37] [38], and for that reason in this work, only the major aspects will be presented. First is assumed that a random process and its observation can be expressed by (3.1) and (3.2) respectively. Where k denotes a time instant t_k, $\mathbf{x}_k$ is the $(n \times 1)$ state vector at instant k, A_k is the $(n \times n)$ transition matrix, C_k the $(m \times n)$ observation matrix, and $\mathbf{z}_k$ the $(m \times 1)$ vector of measurements at instant t_k.

$$\mathbf{x}_{k+1} = A_k \mathbf{x}_k + \mathbf{w}_k \tag{3.1}$$

$$\mathbf{z}_k = C\mathbf{x}_k + \mathbf{v}_k \tag{3.2}$$

Also, $\mathbf{w}_k$ is the $(n \times 1)$ input white noise contribution to the state vector, and $\mathbf{v}_k$ is the $(m \times 1)$ measurement error assumed to be a white sequence. For both vectors is assumed that their covariances are known, and the covariance matrices given by (3.3)-(3.5).

$$E\left[\mathbf{w}_k \mathbf{w}_i^T\right] = \begin{cases} Q_k, & i = k \\ 0, & i \neq k \end{cases} \tag{3.3}$$

$$E\left[\mathbf{v}_k \mathbf{v}_i^T\right] = \begin{cases} R_k, & i = k \\ 0, & i \neq k \end{cases} \tag{3.4}$$

$$E\left[\mathbf{w}_k \mathbf{v}_i^T\right] = 0, \quad i = k \tag{3.5}$$

Is now defined $\hat{\mathbf{x}}_k^-$ as the estimation of the state vector for the instant t_k done in the instant t_{k-1}. Is also assumed that the *a priori* error covariance matrix is known and defined by (3.6) where the estimation error is (3.7).

$$P_k^- = E\left[\mathbf{e}_k^- \mathbf{e}_k^{-T}\right] \tag{3.6}$$

$$\mathbf{e}_k^- = \mathbf{x}_k - \hat{\mathbf{x}}_k^- \tag{3.7}$$

To improve the estimate made in the previous instant $\hat{\mathbf{x}}_k^-$, the measurement $\mathbf{z}_k$ is used. The update estimate for the actual instant is performed using (3.8) where K_k is the Kalman gain defined by (3.9). The deduction to achieve the Kalman gain can be found in [36].

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k\left(\mathbf{z}_k - C_k\hat{\mathbf{x}}_k^-\right) \tag{3.8}$$

$$K_k = P_k^- C_k^T\left(C_k P_k^- C_k^T + R_k\right)^{-1} \tag{3.9}$$

The covariance matrix can be computed using (3.10) for any gain, and in a reduced form using (3.11) for optimal gain condition. Once again is possible to find the deduction in [36].

$$P_k = \left(I - K_k C_k\right) P_k^- \left(I - K_k C_k\right)^T + K_k R_k K_k^T \tag{3.10}$$

$$P_k = \left(I - K_k C_k\right) P_k^- \tag{3.11}$$

The updated estimate for the actual instant ($\hat{\mathbf{x}}_k$) can be easily projected ahead by (3.12) using the transition matrix.

$$\hat{\mathbf{x}}_k^- = A_k\hat{\mathbf{x}}_k \tag{3.12}$$

Combining equation (3.12) with the expression for the next instant estimated error (3.13) is fairly easy to obtain the error covariance matrix for the next time instant (3.14).

$$\mathbf{e}_{k+1}^- = \mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1}^- \tag{3.13}$$

$$P_{k+1}^- = E\left[\mathbf{e}_{k+1}^-\mathbf{e}_{k+1}^{-\,T}\right] = A_k P_k A_k^T + Q_k \tag{3.14}$$

The Kalman Filter recursive algorithm can be summarized and implemented using equations (3.8), (3.9), (3.11) and (3.14), as shown in Fig.3.2. In order to initialize this cycle is necessary to provide, besides the measurements, the initial estimate $\hat{\mathbf{x}}_0^-$, and the initial error covariance matrix P_0^-.

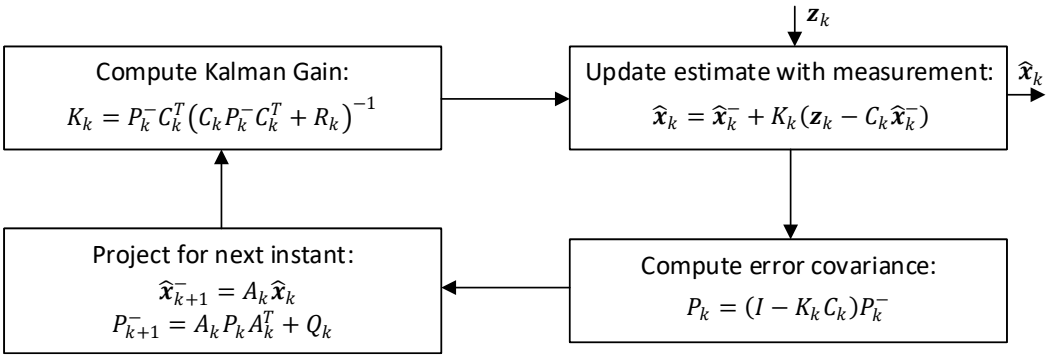

Figure 3.2: Recursive Kalman Filter.

3.2 Discrete Extended Kalman Filter

The discrete Extended Kalman Filter (EKF) is the non-linear version of the already presented Kalman Filter, and like this one, it has been widely explored and documented [36] [39] [40], and for the same reason as the previous, only the major aspects and equations are exposed. Most of the equations and algorithm of the EKF is similar to the KF, with the only difference that the system equations are non-linear.

First is assumed that the discrete process is represented by (3.15), where $\mathbf{x}_k$ is the $(n \times 1)$ state vector that contains the state variables at the instant k, $\mathbf{u}_k$ is the $(m \times 1)$ input vector, $\mathbf{w}_k$ is the process noise contribution to the state vector, and $f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1})$ are the $(n \times 1)$ non-linear process equations that make the transition for instant $k-1$ to k. The observations are given by (3.16), where $\mathbf{z}_k$ is the $(m \times 1)$ vector of observations at the instant k, $\mathbf{v}_k$ is the $(m \times 1)$ noise associated to each measurement, and $h(\mathbf{x}_k)$ are the non-linear equations for the measurements.

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \mathbf{w}_k \tag{3.15}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{3.16}$$

The major difference between the EKF and the KF is that the non-linear equations invalidate the use of state transition matrices and observation matrix as seen in section 3.1. To deal with these non-linear equations, the EKF uses the Jacobian matrices F_k and H_k, defined by (3.17), from the process and observation equations respectively. These matrices are composed of the partial derivatives of each function to each state variable at the instant k.

$$F_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}} \quad , \quad H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{\mathbf{x}}_k^-} \tag{3.17}$$

Like the linear Kalman Filter, the first step of the recursive equations is the prediction. The states for the new time instant k are computed by 3.18, using the non-linear equations with the information of the previous instant $\hat{\mathbf{x}}_{k-1}$ and the inputs of the actual instant. The predicted error covariance matrix for the instant k is given by (3.19), where Q_{k-1} is defined in the same way as the KF by (3.3).

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}) \tag{3.18}$$

$$P_k^- = F_{k-1} P_{k-1} F_{k-1}^T + Q_{k-1} \tag{3.19}$$

Next are the update equations, where the Kalman gain K_k is similar to the linear version and given by (3.20) where the only difference is the use of the Jacobian matrix of the observation equations H_k, instead of the C matrix for the linear version.

$$K_k = P_k^- H_k^T \left(H_k^T P_k^- H_k^T + R_k\right)^{-1} \tag{3.20}$$

The state correction (3.21) is also made in the same way as before, with the difference that the observations of the process states is not achieved by $C.\hat{\mathbf{x}}_k$, but from the non-linear equations, $h\left(\hat{\mathbf{x}}_k^-\right)$.

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k\left(\mathbf{z}_k - h\left(\hat{\mathbf{x}}_k^-\right)\right) \tag{3.21}$$

$$P_k = \left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T + K_k R_k K_k^T \tag{3.22}$$

The update covariance matrix (3.22), is the major difference from the linear KF. The equation (3.11), can still be used instead of (3.22), but due to the known issue of numeric instability associated with the EKF, is recommended [36] that this version is used instead, since it can better preserve the symmetry and positive definiteness of the P matrix.

3.3 Attitude Complementary Filter

The attitude complementary filter presented where is an adaptation of the one seen in [19], that combines the heading measurements of a compass, with the angular velocity readings of a gyroscope. Let ψ define the vehicle heading or yaw angle, and $\dot{\psi}$ the angular velocity of yaw, or yaw rate. Using now the discrete equivalent, where k defines an instant in time $t = kT$, and T is the sampling time interval, is then possible to write the discrete equation (3.23) for the yaw angle.

$$\psi_{k+1} = \psi_k + T\dot{\psi}_k \tag{3.23}$$

The yaw rate is given by the gyroscope, which is assumed to be coincident with the vehicle frame, or that some necessary transformations were already made. This measure is defined as corrupted with $\mathbf{w}_{\dot{\psi}k}$ zero-mean Gaussian white-noise, and a constant sensor bias $b_{\dot{\psi}k}$ driven by zero-mean Gaussian white-noise $\mathbf{w}_{b_{\dot{\psi}}k}$.

$$\dot{\psi}_{rk} = \dot{\psi}_k + b_{\dot{\psi}k} + \mathbf{w}_{\dot{\psi}k} \tag{3.24}$$

$$b_{\dot{\psi}k+1} = b_{\dot{\psi}k} + \mathbf{w}_{b_{\dot{\psi}}k} \tag{3.25}$$

The yaw rate reading $\dot{\psi}_{rk}$ from the gyroscope is then given by (3.24), and the yaw rate bias by (3.25). Rewriting the equations (3.23)-(3.25) in a state space form results in system (3.26).

$$\begin{bmatrix} \psi_{k+1} \\ b_{\dot{\psi}k+1} \end{bmatrix} = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \psi_k \\ b_{\dot{\psi}k} \end{bmatrix} + \begin{bmatrix} T \\ 0 \end{bmatrix} \dot{\psi}_{rk} + \begin{bmatrix} -T & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\dot{\psi}k} \\ \mathbf{w}_{b_{\dot{\psi}}k} \end{bmatrix} \tag{3.26}$$

From (3.26), is then possible to write the close-loop system used for the heading and yaw rate bias estimation as (3.27). Where $\hat{y}_{\psi k}$ is the heading estimation in the previous time instant for the current

one, $\hat{\psi}_k$ is the sensor reading which is corrupted with zero-mean Gaussian white-noise $\mathbf{v}_k$.

$$\begin{bmatrix} \hat{\psi}_{k+1} \\ \hat{b}_{\dot{\psi}k+1} \end{bmatrix} = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{\psi}_k \\ \hat{b}_{\dot{\psi}k} \end{bmatrix} + \begin{bmatrix} T \\ 0 \end{bmatrix} \dot{\psi}_{rk} + \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} (y_{\psi k} - \hat{y}_{\psi k}) \tag{3.27a}$$

$$\hat{y}_{\psi k} = \hat{\psi}_k \quad , \quad y_{\psi k} = \psi_{rk} + \mathbf{v}_{\psi k} \tag{3.27b}$$

The feedback gains K_1 and K_2 are the Kalman gains computed by the recursive equations of section 3.1 using the linear time invariant (LTI) system (3.28a).

$$\begin{bmatrix} \psi_{k+1} \\ b_{\dot{\psi}k+1} \end{bmatrix} = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \psi_k \\ b_{\dot{\psi}k} \end{bmatrix} + \begin{bmatrix} -T & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\dot{\psi}k} \\ \mathbf{w}_{b_{\dot{\psi}k}} \end{bmatrix} \tag{3.28a}$$

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \psi_k \\ b_{\dot{\psi}k} \end{bmatrix} + \mathbf{v}_{\psi k} \tag{3.28b}$$

As proven by [19], it can easily be verified that the proposed filter is Uniformly Asymptotically Stable (UAS) [41].

3.4 Position Complementary Filter

The Position Complementary Filter (PCF) is also an adaptation of [19]), which uses the position and acceleration measurements to estimate the vehicle longitudinal and transversal velocities. The readings for this filter come from sensors which use different coordinate systems. The position come from a GPS which has a global reference frame or Earth reference frame $\{E\}$, while the accelerations come from an accelerometer on-board the vehicle which uses the body reference frame $\{B\}$. Also, the needed velocity components estimate should be in the body frame. The main difference from the filter used by [19]), is the introduction of the accelerometer bias estimate.

Since the proposed estimator architecture uses planar movement only the x and y components are needed unlike the PCF proposed by [19]. Three vectors are defined, the position vector $\bar{\mathbf{p}} = [x \quad y]^T$, the velocity vector $\bar{\mathbf{v}} = [v_x \quad v_y]^T$, and the acceleration vector $\bar{\mathbf{a}} = [a_x \quad a_y]^T$. It is also defined R as the rotation matrix that transform body frame, into earth frame coordinates such as (3.29), where ψ is the heading of the vehicle.

$${}^E\bar{\mathbf{v}} = R_\psi {}^B\bar{\mathbf{v}}; \quad R_\psi = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \tag{3.29}$$

Considering the continuous-time kinematics $\dot{\bar{\mathbf{p}}} = \bar{\mathbf{v}}$, $\dot{\bar{\mathbf{v}}} = \bar{\mathbf{a}}$, is then possible to derive the discrete-time equivalent that represents the equations of motion as (3.30), where $R_{\psi k} = R_\psi(\psi_k)$ represents the rotation matrix with a yaw angle ψ at the instant k, and T is the time sample.

$$\bar{\mathbf{p}}_{k+1} = \bar{\mathbf{p}}_k + T\bar{\mathbf{v}}_k + \frac{T^2}{2} R_k {}^B\bar{\mathbf{a}}_k \tag{3.30a}$$

$$\bar{\mathbf{v}}_{k+1} = \bar{\mathbf{v}}_k + T R_{\psi k} {}^B\bar{\mathbf{a}}_k \tag{3.30b}$$

In the same way as seen for the gyroscope in the ACF, the accelerometers often have a bias in the readings. Analogous to (3.24)-(3.25) the accelerometer model and bias are (3.31) and (3.32), where $\bar{\mathbf{w}}_{a_r k}$ is a vector of zero-mean, Gaussian white noise, and the vector of bias (for both axis) $\bar{\mathbf{b}}_{ak}$ is also driven by a vector of Gaussian white noise $\bar{\mathbf{w}}_{b_a k}$.

$$\bar{\mathbf{a}}_{rk} = \bar{\mathbf{a}}_k + \bar{\mathbf{b}}_{ak} + \bar{\mathbf{w}}_{a_r k} \tag{3.31}$$

$$\bar{\mathbf{b}}_{ak+1} = \bar{\mathbf{b}}_{ak} + \bar{\mathbf{w}}_{b_a k} \tag{3.32}$$

Rewriting and combining equations (3.30)-(3.32) in a state space form results in the system (3.33), where I is a 2x2 identity matrix, and $\bar{\mathbf{w}}_{pk}$ is the noise associated to the position.

$$\begin{bmatrix} \bar{\mathbf{p}}_{k+1} \\ \bar{\mathbf{v}}_{k+1} \\ \bar{\mathbf{b}}_{ak+1} \end{bmatrix} = \begin{bmatrix} I & TR & -\frac{T^2}{2}R \\ 0 & I & -TI \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}}_k \\ \bar{\mathbf{v}}_k \\ \bar{\mathbf{b}}_{ak} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2}R \\ TI \\ 0 \end{bmatrix} \bar{\mathbf{a}}_{rk} + \begin{bmatrix} I & -\frac{T^2}{2}I & 0 \\ 0 & -TI & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \bar{\mathbf{w}}_{pk} \\ \bar{\mathbf{w}}_{ak} \\ \bar{\mathbf{w}}_{b_a k} \end{bmatrix} \tag{3.33}$$

From (3.33) is then possible to write the close-loop system (3.34) for the velocity components estimate. Where $\mathbf{y}_{pk}$ is the vector with the x and y coordinates given by the GPS, $\hat{\mathbf{y}}_{pk}$ is position estimation on the previous time instance for the actual, and $\bar{\mathbf{v}}_{pk}$ accounts for the small perturbations in the measurements of the location system.

$$\begin{bmatrix} \bar{\mathbf{p}}_{k+1} \\ \bar{\mathbf{v}}_{k+1} \\ \bar{\mathbf{b}}_{ak+1} \end{bmatrix} = \begin{bmatrix} I & TR & -\frac{T^2}{2}R \\ 0 & I & -TI \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}}_k \\ \bar{\mathbf{v}}_k \\ \bar{\mathbf{b}}_{ak} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2}R \\ TI \\ 0 \end{bmatrix} \bar{\mathbf{a}}_{rk} + \begin{bmatrix} K_1 \\ R_k^T K_2 \\ R_k^T K_3 \end{bmatrix} (\mathbf{y}_{pk} - \hat{\mathbf{y}}_{pk}) \tag{3.34a}$$

$$\hat{\mathbf{y}}_{pk} = \hat{\mathbf{p}}_k \quad , \quad \mathbf{y}_{pk} = \bar{\mathbf{p}}_k + \bar{\mathbf{v}}_{pk}. \tag{3.34b}$$

The gains K_1, K_2 and K_3 are 2x2 diagonal matrices with the Kalman gains computed using the recursive algorithm presented in section 3.1. To attain these Kalman gains, a time invariant system is defined as (3.35), which is the special case when yaw is zero ($\psi = 0$).

$$\begin{bmatrix} \bar{\mathbf{p}}_{k+1} \\ \bar{\mathbf{v}}_{k+1} \\ \bar{\mathbf{b}}_{ak+1} \end{bmatrix} = \begin{bmatrix} I & TI & -\frac{T^2}{2}I \\ 0 & I & -TI \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}}_k \\ \bar{\mathbf{v}}_k \\ \bar{\mathbf{b}}_{ak} \end{bmatrix} + \begin{bmatrix} I & -\frac{T^2}{2}I & 0 \\ 0 & -TI & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \bar{\mathbf{w}}_{pk} \\ \bar{\mathbf{w}}_{ak} \\ \bar{\mathbf{w}}_{b_a k} \end{bmatrix} \tag{3.35a}$$

$$y_{xk} = \begin{bmatrix} I & 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}}_k \\ \bar{\mathbf{v}}_k \\ \bar{\mathbf{b}}_{ak} \end{bmatrix} + \bar{\mathbf{v}}_{pk} \tag{3.35b}$$

As proven by [19] for the filter without bias, it can easily be verified that the proposed filter (3.34) is Uniformly Asymptotically Stable (UAS) [41].

3.5 Vehicle Linear Estimator

The Vehicle Linear Estimator (VLE) arises from the need to introduce the vehicle dynamics to the sideslip estimator architecture, since both complementary filters only use particle cinematic equations. The proposed estimator uses the linear car model presented in section 2.4, which relies also on a linear tyre approximation also presented before in section 2.1.3.

The vehicle linear estimator is based on the (2.26) which is time-varying depending on the longitudinal velocity estimate $\hat{v}_x$ from the PCF. It has as state variables the lateral velocity estimate $\hat{v}_y$ also from the PCF and the bias corrected yaw rate. Thus, based on system (2.26), the underlying model for the estimator design is expressed as (3.36a).

$$\begin{bmatrix} \dot{v}_y \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} -\frac{C_{\alpha f}+C_{\alpha r}}{mv_x} & \frac{-aC_{\alpha f}+bC_{\alpha r}}{mv_x} - v_x \\ \frac{-aC_{\alpha f}+bC_{\alpha r}}{I_\psi v_x} & -\frac{a^2C_{\alpha f}+b^2C_{\alpha r}}{I_\psi v_x} \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \frac{C_{\alpha f}}{m} \\ \frac{aC_{\alpha f}}{I_\psi} \end{bmatrix} \delta_r + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{w}_{v_y} \\ \mathbf{w}_{\dot{\psi}} \end{bmatrix} \tag{3.36a}$$

$$y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \mathbf{v}_{v_y} \\ \mathbf{v}_{\dot{\psi}} \end{bmatrix} \tag{3.36b}$$

On the dynamics of this system, an important detail should be noted. Due to the dependency of the longitudinal velocity, if the car is stopped, every entry of the transition matrix is dividing by zero, causing the system to fail. Also, using longitudinal velocity values to close to zero will result in numerical problems.

The (3.36a) system is continuous. The main objective of this document is to find an estimator architecture that can provide the sideslip of the vehicle on-board. And for making this implementation is convenient that the filters are discrete to reduce the computation power needed. In order to make the achieve the discrete model, a property [42] is used where the conversion from continuous to discrete is given by (3.37). Where A_c is the generic continuous transition matrix, B_c is the continuous discrete matrix and A_k and B_k are the respective discrete equivalents for a sampling time T.

$$\exp\left(\begin{bmatrix} A_c & B_c \\ 0 & 0 \end{bmatrix} T\right) = \begin{bmatrix} A_k & B_k \\ 0 & I \end{bmatrix} \tag{3.37}$$

The matrix of the exponential has as many rows of zeros as necessary to assure the matrix is square, and the I in the discrete is the identity matrix that assures the result is a square matrix.

For the proposed filter is introduced the A_k^{vle} and the B_k^{vle}, as the discrete matrices for the system (3.36a).

$$\begin{aligned} \begin{bmatrix} \hat{v}_{yk+1} \\ \hat{\dot{\psi}}_{k+1} \end{bmatrix} &= A_k^{vle} \begin{bmatrix} \hat{v}_{yk} \\ \hat{\dot{\psi}}_k \end{bmatrix} + B_k^{vle}\delta_r + \begin{bmatrix} K_1 & K_2 \\ K_3 & K_4 \end{bmatrix} (y_k - \hat{y}_k) \\ \hat{y}_k = \begin{bmatrix} \hat{v}_{yk} \\ \hat{\dot{\psi}}_k \end{bmatrix}, \quad y_k &= \begin{bmatrix} v_{yk} \\ \dot{\psi}_k \end{bmatrix} + \begin{bmatrix} \mathbf{v}_{v_y} \\ \mathbf{v}_{\dot{\psi}} \end{bmatrix} \end{aligned} \tag{3.38}$$

The time varying filter is then given by (3.38), where $\mathbf{v}_{v_y}$ and $\mathbf{v}_{\dot{\psi}}$ are zero-mean, Gaussian white noise, $\hat{y}$ is the estimated state vector of the linear model, and y is the vector that contains the most recent values of v_y from the Position Complementary Filter and $\dot{\psi}$ from the gyroscope with bias correction. The K_i gains are the Kalman gains that relate the error of measured and estimated data to each state variable computed using the recursive equations presented in section 3.1.

For this filter it is not possible to directly prove its stability since it is not an LTI system due to the longitudinal velocity. However, if a value of longitudinal velocity is fixed, using the Lyapunov stability criterions [41] is possible to prove its stability.

3.6 Vehicle Non-Linear Estimator

Previously, the VLE was presented with the intent of introducing the car dynamics in the sideslip estimator architecture. Has explained before in section 2.4, the linear vehicle model has several limitations, being the most relevant, the small angle approximation, and the tyre model. To overcome these limitations, is presented the Vehicle Non-Linear Estimator (VNLE) as an option to the VLE. This estimator is not limited to small angles and uses the Burckhardt tyre model, making it closer to reality than the VLE. On the other hand, since the equations are non-linear, is necessary to use the EKF already presented in section 3.2, which is computationally heavier and may have some numerical problems.

This estimator uses the non-linear planar model equations (2.16) as the process equation. Since the main objective is always to have real-time estimation, the filter should be in discrete-time to facilitate the implementation in an on-board system. With this is mind (2.16) is discretized using Euler method, where the accelerations of (2.16) are replaced by (3.39), and T is the sampling time, or time step between two consecutive instants.

$$\dot{v}_x = \frac{v_{x_{k+1}} - v_{x_k}}{T}\ , \quad \dot{v}_y = \frac{v_{y_{k+1}} - v_{y_k}}{T}\ , \quad \ddot{\psi} = \frac{\dot{\psi}_{k+1} - \dot{\psi}_k}{T} \tag{3.39}$$

It is known that the Euler method induces some error at each iteration, but since the time sampling interval will be small, around 0.01 seconds, and the equations will work in a close loop, the error associated with this method is of little concern. The discrete version of the non-linear planar model is then given by (3.40).

$$f_1:\ v_{x_{k+1}} = v_{x_k} + v_y\dot{\psi}T - \frac{1}{m}[F_y^F \sin\delta - F_x^F \cos\delta - F_x^R]T \tag{3.40a}$$

$$f_2:\ v_{y_{k+1}} = v_{y_k} - v_x\dot{\psi}T + \frac{1}{m}[F_y^F \cos\delta + F_y^R + F_x^F \sin\delta]T \tag{3.40b}$$

$$f_3:\ \dot{\psi}_{k+1} = \dot{\psi}_k + \frac{1}{I_\psi}a[F_y^F \cos\delta + F_x^F \sin\delta]T - \frac{1}{I_\psi}bF_y^R T \tag{3.40c}$$

$$F_k = \begin{bmatrix} \left.\frac{\partial f_1}{\partial v_x}\right|_{\hat{x}_k,u_k} & \left.\frac{\partial f_1}{\partial v_y}\right|_{\hat{x}_k,u_k} & \left.\frac{\partial f_1}{\partial \dot{\psi}}\right|_{\hat{x}_k,u_k} \\ \left.\frac{\partial f_2}{\partial v_x}\right|_{\hat{x}_k,u_k} & \left.\frac{\partial f_2}{\partial v_y}\right|_{\hat{x}_k,u_k} & \left.\frac{\partial f_2}{\partial \dot{\psi}}\right|_{\hat{x}_k,u_k} \\ \left.\frac{\partial f_3}{\partial v_x}\right|_{\hat{x}_k,u_k} & \left.\frac{\partial f_3}{\partial v_y}\right|_{\hat{x}_k,u_k} & \left.\frac{\partial f_3}{\partial \dot{\psi}}\right|_{\hat{x}_k,u_k} \end{bmatrix} \tag{3.41}$$

In order to implement the EKF, it is necessary to compute the Jacobian matrices. The F_k matrix is given by (3.41), and the functions are (3.40). The partial derivatives of each function are expressed by equations (3.42).

$$\frac{\partial f_1}{\partial v_x} = -\frac{T\sin\delta}{m}\left[\frac{\partial F_y^{FL}}{\partial v_x} + \frac{\partial F_y^{FR}}{\partial v_x}\right] + 1 \tag{3.42a}$$

$$\frac{\partial f_1}{\partial v_y} = -\frac{T\sin\delta}{m}\left[\frac{\partial F_y^{FL}}{\partial v_y} + \frac{\partial F_y^{FR}}{\partial v_y}\right] + rT \tag{3.42b}$$

$$\frac{\partial f_1}{\partial \dot{\psi}} = -\frac{T\sin\delta}{m}\left[\frac{\partial F_y^{FL}}{\partial \dot{\psi}} + \frac{\partial F_y^{FR}}{\partial \dot{\psi}}\right] + v_y T \tag{3.42c}$$

$$\frac{\partial f_2}{\partial v_x} = \frac{T\cos\delta}{m}\left[\frac{\partial F_y^{FL}}{\partial v_x} + \frac{\partial F_y^{FR}}{\partial v_x}\right] + \frac{T}{m}\left[\frac{\partial F_y^{RL}}{\partial v_x} + \frac{\partial F_y^{RR}}{\partial v_x}\right] - \dot{\psi}T \tag{3.42d}$$

$$\frac{\partial f_2}{\partial v_y} = \frac{T\cos\delta}{m}\left[\frac{\partial F_y^{FL}}{\partial v_y} + \frac{\partial F_y^{FR}}{\partial v_y}\right] + \frac{T}{m}\left[\frac{\partial F_y^{RL}}{\partial v_y} + \frac{\partial F_y^{RR}}{\partial v_y}\right] + 1 \tag{3.42e}$$

$$\frac{\partial f_2}{\partial \dot{\psi}} = \frac{T\cos\delta}{m}\left[\frac{\partial F_y^{FL}}{\partial \dot{\psi}} + \frac{\partial F_y^{FR}}{\partial \dot{\psi}}\right] + \frac{T}{m}\left[\frac{\partial F_y^{RL}}{\partial \dot{\psi}} + \frac{\partial F_y^{RR}}{\partial \dot{\psi}}\right] - v_x T \tag{3.42f}$$

$$\frac{\partial f_3}{\partial v_x} = \frac{aT\cos\delta}{I_\psi}\left[\frac{\partial F_y^{FL}}{\partial v_x} + \frac{\partial F_y^{FR}}{\partial v_x}\right] - \frac{bT}{I_\psi}\left[\frac{\partial F_y^{RL}}{\partial v_x} + \frac{\partial F_y^{RR}}{\partial v_x}\right] \tag{3.42g}$$

$$\frac{\partial f_3}{\partial v_y} = \frac{aT\cos\delta}{I_\psi}\left[\frac{\partial F_y^{FL}}{\partial v_y} + \frac{\partial F_y^{FR}}{\partial v_y}\right] - \frac{bT}{I_\psi}\left[\frac{\partial F_y^{RL}}{\partial v_y} + \frac{\partial F_y^{RR}}{\partial v_y}\right] \tag{3.42h}$$

$$\frac{\partial f_3}{\partial \dot{\psi}} = \frac{aT\cos\delta}{I_\psi}\left[\frac{\partial F_y^{FL}}{\partial \dot{\psi}} + \frac{\partial F_y^{FR}}{\partial \dot{\psi}}\right] - \frac{bT}{I_\psi}\left[\frac{\partial F_y^{RL}}{\partial \dot{\psi}} + \frac{\partial F_y^{RR}}{\partial \dot{\psi}}\right] + 1 \tag{3.42i}$$

The above equations, depend on the partial derivative of the lateral force given by the Burkhardt tyre model (3.43). Since the difference between front, rear, left and right wheels is only visible in the slip angle (α_i), a generic equation (3.44) is used at this point. The same goes for the state variables, that are only present in the slip, and for that reason is introduced a generic term $\Gamma_i = [v_x\ v_y\ r]$ to represent the 3 partial derivatives.

$$F_{y_i} = \left(c_1\left(1 - e^{-c_2 s_{r_i}}\right) - c_3 s_{r_i}\right)\left(1 - c_5 F_{z_i}^2\right) F_{z_i} \tag{3.43}$$

$$\frac{\partial F_{y_i}}{\partial \Gamma_i} = \left(c_1 - \left(c_1 e^{-c_2 s_{r_i}}\right)\left(-c_2\frac{\partial s_{r_i}}{\partial \Gamma_i}\right) - c_3\frac{\partial s_{r_i}}{\partial \Gamma_i}\right)\left(1 - c_5 F_{z_i}^2\right) F_{z_i} \tag{3.44}$$

The tyre model uses a definition of resulting slip as explained in section 2.1.1. Combining Tab.2.1 with (2.1), results in (3.45). Once again is used the simplification of generic derivative Γ_i, resulting in (3.46).

$$s_{r_i} = |\tan(\alpha_i)| \tag{3.45}$$

$$\frac{\partial s_{r_i}}{\partial \Gamma_i} = \frac{\partial \alpha_i}{\partial \Gamma_i} \frac{\tan(\alpha_i)\sec^2(\alpha_i)}{|\tan(\alpha_i)|} \tag{3.46}$$

The partial derivative of the wheel slip angle (2.21) to each state variable are given by (3.47)-(3.49). The difference of each wheel will be the location of the wheel relative to the CG given by the vector $[x_i \; y_i]$.

$$\frac{\partial \alpha_i}{\partial v_x} = \frac{-\dfrac{v_y + x_i\dot{\psi}}{\left(v_x - y_i\dot{\psi}\right)^2}}{\left(\dfrac{v_y + x_i\dot{\psi}}{v_x - y_i\dot{\psi}}\right)^2 + 1} \tag{3.47}$$

$$\frac{\partial \alpha_i}{\partial v_x} = \frac{\dfrac{1}{v_x - y_i\dot{\psi}}}{\left(\dfrac{v_y + x_i\dot{\psi}}{v_x - y_i\dot{\psi}}\right)^2 + 1} \tag{3.48}$$

$$\frac{\partial \alpha_i}{\partial v_x} = \frac{-\dfrac{v_y y_i + v_x x_i}{\left(v_x - y_i\dot{\psi}\right)^2}}{\left(\dfrac{v_y + x_i\dot{\psi}}{v_x - y_i\dot{\psi}}\right)^2 + 1} \tag{3.49}$$

The observation equations for this model are simple, since they are the same as the state variables. The Jacobian matrix H_k for the observations is then given by a simple 3-by-3 identity matrix.

For this estimator, the stability is not explored due to its complexity, however some works have already explored the stability criteria necessary for the non-linear Kalman filter like [43].

Chapter 4

Simulation

This chapter presents the simulation using the models of Chapter 2, to test the proposed estimator architecture of Chapter 3.

First is exposed the vehicle used for the tests, which is based on FST06e (Fig.4.1), a real Formula Student prototype. To recreate this car, the non-linear models presented in Chapter 2 are interconnected in order to model as accurate as possible the behaviour of the real car. The states of the different models that make up the vehicle are then combined to reproduce the sensors available in the car, and all the relations between the states. The sensors assumed to be on-board are a GPS, an IMU and a steering encoder. The measurements taken from these sensors are then corrupted with random white noise, uncorrelated between measurements.

In a second part, the estimators are tested in optimal conditions, which means that the vehicle parameters are accurate, and the difference between estimator's model and the vehicle model is minimum.

Figure 4.1: FST06e.

Variable		Value	Unit	Variable		Value	Unit
Car weight + driver	m	350	kg	Susp. front stif (i=1,2)	K_{si}	44692	N/m
Unsprung mass (i=1,2,3,4)	m_{ui}	9.2	kg	Susp. rear stif (i=3,4)	K_{si}	50400	N/m
Dist. CG to front axle	a	0.873	m	Susp. front damp (i=1,2)	C_{si}	2941	Ns/m
Dist. CG to rear axle	b	0.717	m	Susp. rear damp (i=3,4)	C_{si}	4194	Ns/m
Left half track	c	0.6	m	Anti-roll bar stif (i=1,2)	K_{ari}	17513	N/m
Right half track	d	0.6	m	Tyre stiffness (i=1,2,3,4)	K_{pi}	113290	N/m
Moment of inertia, roll	I_ϕ	22.67	kg.m^2	Tyre damping (i=1,2,3,4)	C_{pi}	1436	Ns/m
Moment of inertia, pitch	I_θ	74.62	kg.m^2	Drag coefficient	C_d	1.1	N/(m/s)^2
Moment of inertia, yaw	I_ψ	120.13	kg.m^2	Lift coefficient	C_l	2.13	N/(m/s)^2

Table 4.1: Variables for the different vehicle modules, based on a real Formula Student prototype, to be used in the Simulation chapter.

The results for each estimate are presented, briefly discussed and compared with the perfect signals from the vehicle model. These perfect signals are denoted as the real value.

In a final part, the estimator architecture is tested for erroneous conditions. The estimators are kept equal to the previous test, and some alteration are made to the vehicle model. These alterations are the tyre model, which is changed from dry to wet conditions, and the track is no more perfectly flat.

In both simulation tests, the pros and cons of the linear and the non-linear vehicle estimators are discussed.

It also should be noted that for every filter only the final weights and gains are presented for simplification. To reach these values, the weights started as the variance of the signals, and an adjust was made to improve the results leading to the presented values.

4.1 Simulation variables

4.1.1 Vehicle

The vehicle used for the simulations ahead is based on a real Formula Student prototype, FST06e (Fig.4.1). The values used in the model to simulate the vehicle were given by the team and are summarized in Tab.4.1. The car weights 350kg with a driver and has a 45%-55% (front-rear) weight distribution. The wheelbase of the car is 1.59m with two front and rear tracks of 1.20m. The suspension values for stiffness and damping are also presented as the equivalent of quarter suspension which depends on the suspension geometry. The tyres are modelled with the Burckhardt equation (2.2) and the coefficients are the ones for the fitting of the TTC data presented in section 2.1.2.

On the car two limits are imposed to the inputs. The first is the steering angle, which is limited to a range of -30º to 30º. The second one is the longitudinal force. The car has two 50kW electric PMSM (permanent magnet synchronous motor) motors, and as is characteristic of electric motors, the maximum torque is available right from the 0rpm until the maximum power limit is reached. The car as a single fixed gear with a ratio of 4.1:1, and an effective tyre radius of 0.228m, which gives a maximum longitudinal force of 3.8kN. For these simulations, as explained in section 2.3, the longitudinal force is assumed a direct input on the CG. For braking, the force produced by the brakes is dependent of the velocity and is much higher, but for simplification is used the same limit of force but in reverse of -3.8kN.

4.1.2 Sensors

As seen in section 2.6, to replicate the available data present in the car, a set of sensors was modelled. These ones, were corrupted with random white noise, and some of them have an offset. In Tab.4.2 are represented the statistical values of the errors associated with each sensor. The GPS doesn't have an offset because is irrelevant, since the position is only used to estimate the velocity, a constant offset to the position wouldn't produce any effect.

	Accelerometer [m/s^2]			Gyroscope [º/s]			GPS [m]		Comp.	Steering
	x	y	z	x	y	z	x	y	[º]	enc. [º]
mean	-0.060	0.170	-0.100	1.500	0.300	-1.200	n/a	n/a	n/a	-0.400
σ	0.706	0.709	0.714	0.627	0.645	0.695	0.064	0.064	0.588	0.265
3σ	2.118	2.127	2.142	1.881	1.935	2.085	0.192	0.192	1.764	0.795
σ^2	0.498	0.502	0.393	0.393	0.416	0.483	0.004	0.004	0.346	0.070

Table 4.2: Statistical data for the noise associated to each sensor in the model. Where σ represents the standard deviation, σ^2 the variance, and n/a means not applicable.

4.2 Estimation Results

For the following simulations, the sampling time chosen was of t=0.01s, this because most of the sensors work at these frequencies, around 100Hz, and since the main objective is to provide estimated values for a control strategy, where much lower frequencies would be almost useless.

The inputs for the simulation are a combination of soft and hard forward accelerations and braking, and narrow and wide curves. The performed trajectory can be seen in Fig.4.2. For these first tests, the ground is kept flat (no banking). In section 4.2.4 the influence of the ground will be explored.

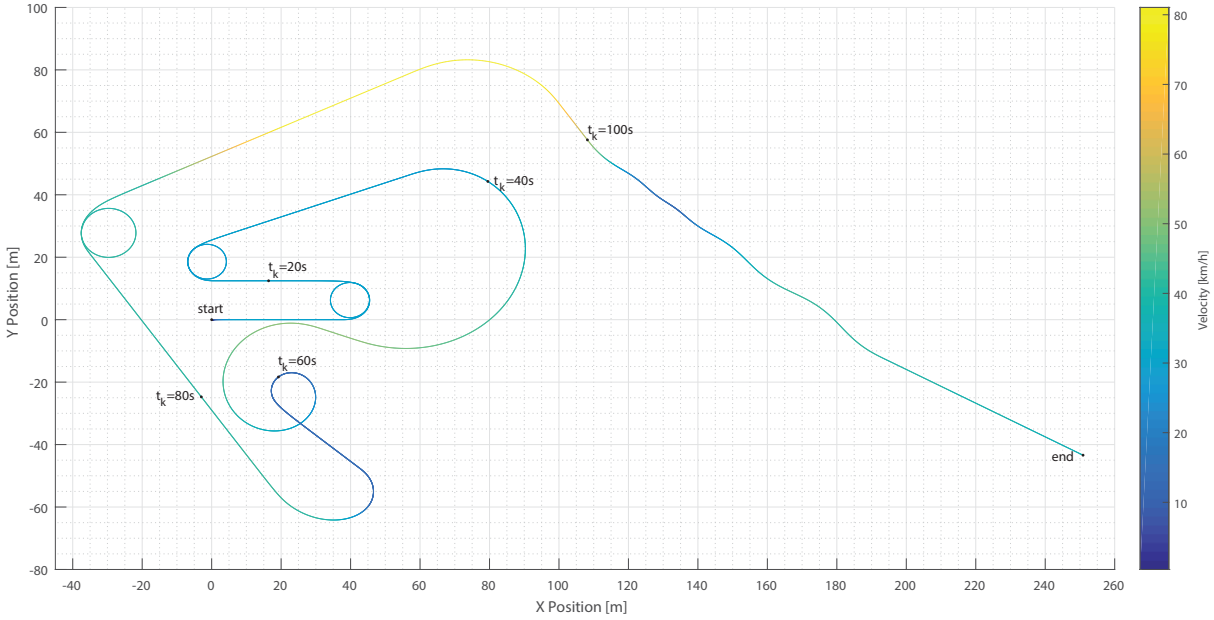

Figure 4.2: Trajectory performed by the vehicle in the present simulation test. Some time markers are present to better relate the path with the remaining time graphics.

4.2.1 Attitude Complementary Filter

The first filter, as seen in Fig.3.1, is for the attitude which combines the yaw from the magnetometer readings, with the yaw rate from the gyroscope. As seen in Tab.4.2, the z-axis of the gyroscope has an initial offset of -1.2deg/s, which can fluctuate due to the contributions of the remain axis during roll and pitch rotations. The filter was presented in section 3.3, and all the filter parameters can be found in Tab.4.3. Since all the variables for this filter are always constant, the Kalman gain was not computed iteratively along the simulation, but "a priori", and the optimal gain found was kept the same along the simulation. In Fig.4.3 is presented the frequency response of the filter. The sum of both transfer functions is unitary, as expected from a complementary filter.

Observation weight	State weight	Filter Gain
$R_{acf} = \begin{bmatrix} 10^3 \end{bmatrix}$	$Q_{acf} = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{bmatrix}$	$K_{acf} = \begin{bmatrix} 4.572 \times 10^{-3} \\ -9.977 \times 10^{-4} \end{bmatrix}$

Table 4.3: Parameters for the Attitude Complementary Filter in simulation.

The initial condition is $\hat{x}_0 = [5,\ 0]^T$. The results of the filter can be seen in Fig.4.4, where is compared the error to the true value, from the magnetometer, and from the ACF. To note that the wrong initial condition of yaw is corrected after 8sec. Also, the yaw rate bias is estimated and can be seen in Fig.4.5. As stated above, the initial condition of the bias is zero, but several tests were made with different values to show the convergence of the estimated bias. Once again, after around 20sec, the estimated yaw rate bias converges around the offset assumed for the gyroscope z-axis in Tab.4.2. The oscillations around this offset are due to the roll and pitch.

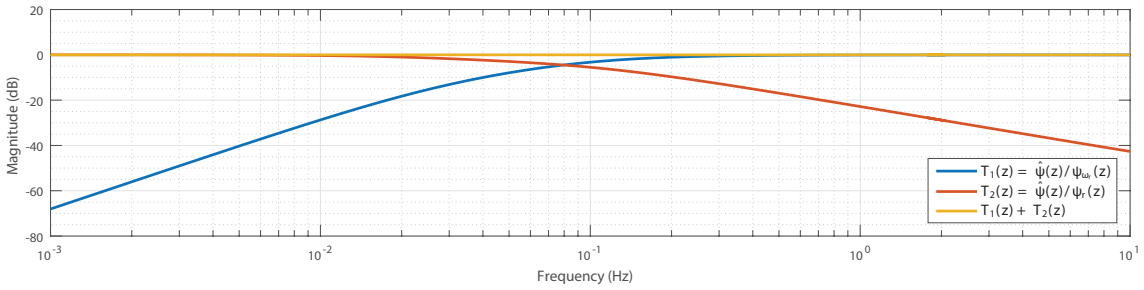

Figure 4.3: Attitude complementary filter transfer function.

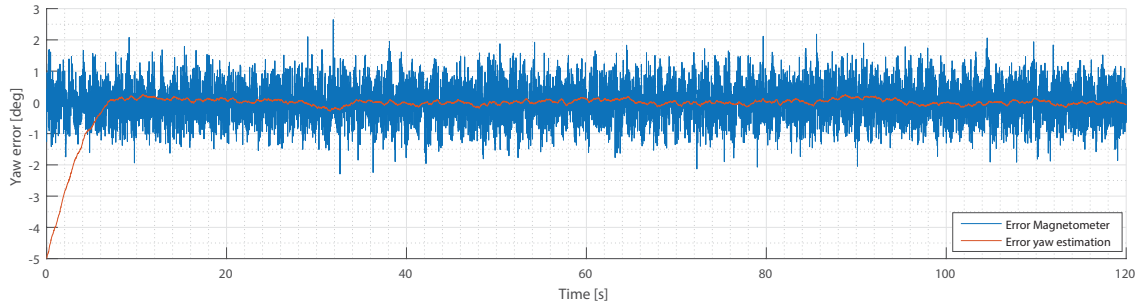

Figure 4.4: Yaw error comparison between the magnetometer and the ACF result relative to the real yaw angle of the vehicle.

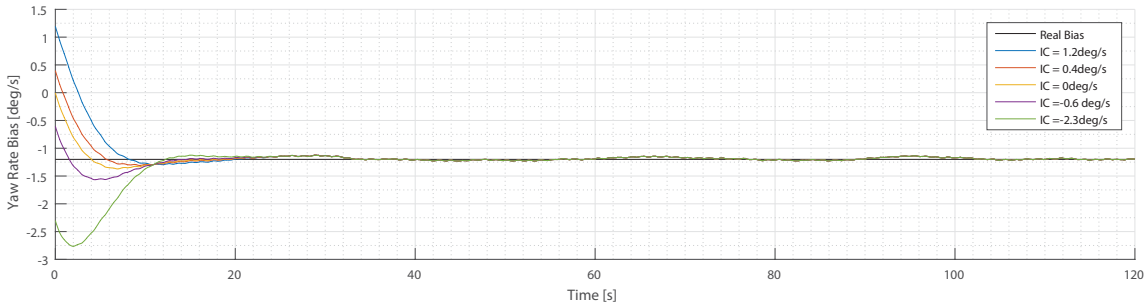

Figure 4.5: Convergence of yaw rate bias estimate from the ACF, for different initial conditions (IC).

4.2.2 Position Complementary Filter

The Position Complementary Filter besides the position as an observation and the accelerations from the IMU as and input, uses also the estimated yaw angle inside the filter matrices. As explained in section 3.4, the filter gains are computed using the time-invariant equation (3.35). The parameters for this gain can be found in Tab.4.4, and the frequency response of the close-loop filter is depicted in Fig.4.6, which shows a unitary gain for the sum of both transfer functions as expected of a complementary filter.

In order to demonstrate the influence of the estimated bias in the accelerations, to PCF filters were tested, one with the bias as seen in section 3.4 and other without the bias. This last one (without bias) is used only for this comparison.

The initial conditions of the filter were $\hat{x}_0 = [0, 0, 3, -0.5, 0, 0]^T$. The results of the longitudinal velocity can be seen in Fig.4.7 and the estimated lateral velocity in Fig.4.8.

For the longitudinal velocity, little can be said. Both filters have satisfactory results, and can recover under 1 second from a wrong initial condition. The difference to the real velocity is almost imperceptible

Observation weight	$R_{pcf} = \begin{bmatrix} 5 \times 10^1 & 0 \\ 0 & 5 \times 10^1 \end{bmatrix}$
State weight	$Q_{pcf} = \begin{bmatrix} 10^{-3}\,\mathbf{I}_2 & 0 & 0 \\ 0 & \mathbf{I}_2 & 0 \\ 0 & 0 & 2 \times 10^{-2}\,\mathbf{I}_2 \end{bmatrix}$
Filter Gain	$K_{pcf} = \begin{bmatrix} 5.33 \times 10^{-2}\,\mathbf{I}_2 & 0 & 0 \\ 0 & 1.45 \times 10^{-1}\mathbf{I}_2 & 0 \\ 0 & 0 & 1.95 \times 10^{-2}\,\mathbf{I}_2 \end{bmatrix}$

Table 4.4: Parameters for the Position Complementary Filter in simulation, where $\mathbf{I}_2$ denotes a 2×2 identity matrix

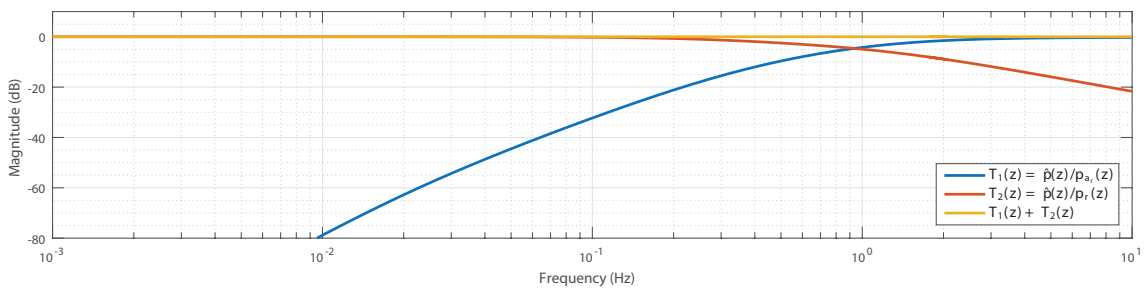

Figure 4.6: Position complementary filter transfer function.

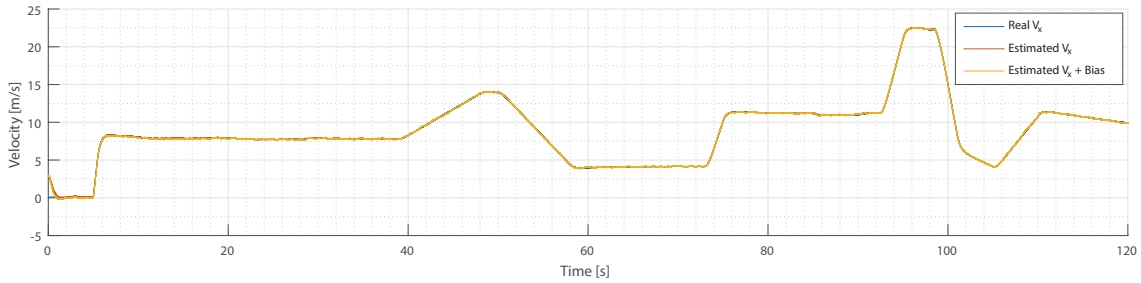

Figure 4.7: Estimated longitudinal velocity compared with the real. Two position complementary filters were tested, one with acceleration bias estimated, other without.

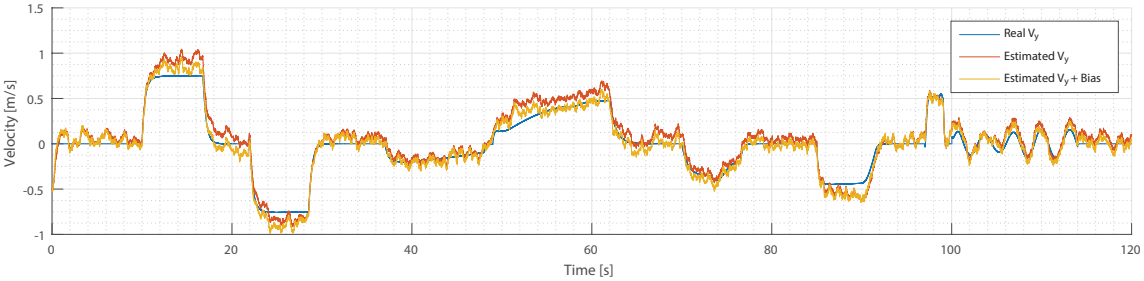

Figure 4.8: Estimated lateral velocity compared with the real. Two position complementary filters were tested, one with acceleration bias estimated, other without.

due to the scale of values and the effect of the bias is irrelevant in this case. The error to the real velocity can be seen in Tab.4.5.

In the lateral velocity estimate, the errors are more perceptible, largely due to the range of lateral velocities achieved. Compared with the error of the longitudinal velocity (Tab.4.5), the standard deviation to the real value is greater, but the effect of the bias is evident whether it is in statistical values (Tab.4.5) or in graphical data (Fig.4.8). The estimate using the bias, shows an advantage since the results are closer to real, and since for lateral velocities close to zero, the estimate does not have an offset.

The acceleration bias, unlike the yaw rate bias estimated in section 4.2.1, does not depend only on the accelerometer offset, but a combination of the 3-axis offset of the accelerometer and gyroscope and the rotation induce by roll and pitch, given by the accelerometer model (2.34). Due to this, isn't possible to expect a convergence to a constant value, instead is possible to evaluate the bias by the convergence from different initial conditions as seen in Fig.4.9 and Fig.4.10. In the longitudinal bias (Fig.4.9), the peak before the 5 second mark is due to the initial condition of the longitudinal velocity. This can be verified by setting this condition to match the real one.

	Longitudinal velocity			Lateral velocity		
	mean	σ	3σ	mean	σ	3σ
PCF without bias	0.013	0.053	0.158	0.056	0.079	0.237
PCF with bias	-0.013	0.054	0.162	-0.001	0.071	0.213

Table 4.5: Statistical error of two different PCF estimators relative to the real velocity components.

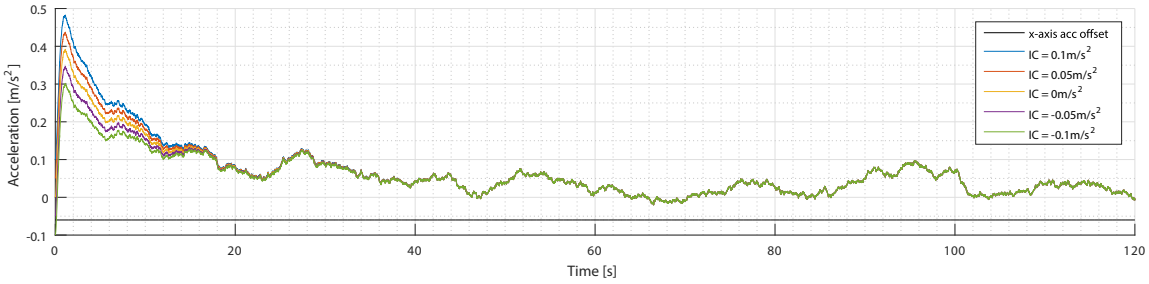

Figure 4.9: Estimated longitudinal acceleration bias for different initial conditions (IC) and the x-axis accelerometer offset.

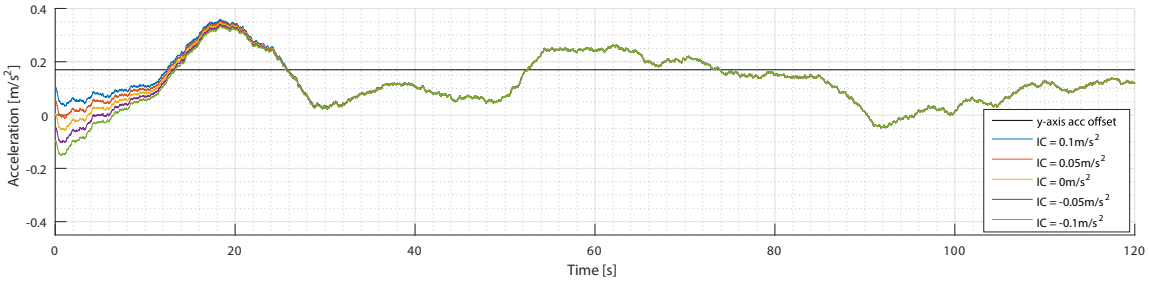

Figure 4.10: Estimated lateral acceleration bias for different initial conditions (IC) and the y-axis accelerometer offset.

4.2.3 Vehicle Estimator

The third estimator as seen in Fig.3.1 introduces the vehicle dynamics to the estimate. For this place two estimators are compared, the Vehicle Linear Estimator (VLE) and the Vehicle Non-Linear Estimator (VNLE), where the objective is to explore the pros and cons of each one. For the final estimator architecture only one is necessary.

Both filters are projected with the same values as the simulation vehicle in section 4.1. The needed variables for the two filters are summed up in Tab.4.6. The cornering stiffness values were computed as explained in section 2.1.3, for the Burckhardt tyre curve using the TTC data, for a slip angle α_i=2º with a vertical load of 773N for the front tyres and 944N for the rear. The tyre load is computed from the 45%-55% weight distribution implicit by the CG position.

For the VLE presented in section 3.5, the filter gains are computed using the Discrete Kalman Gain

Variable		Value	Unit
Car + driver mass	m	350	kg
Moment of inertia, yaw	I_ψ	120.13	kg.m^2
Distance from CG to front axle	a	0.873	m
Distance from CG to rear axle	b	0.717	m
Distance from CG to left wheels	c	0.6	m
Distance from CG to right wheels	d	0.6	m
Cornering Stiffness, front (1 wheel)	$C_{\alpha f_i}$	30170	N/rad
Cornering Stiffness, rear (1 wheel)	$C_{\alpha r_i}$	34820	N/rad
Vertical load, front (1 wheel)	F_{zf}	773	N
Vertical load, rear (1 wheel)	F_{zr}	944	N

Table 4.6: Parameters for the VLE and VNLE estimators. The cornering stiffness approximation was made for Fz=773N at front and Fz=944N at rear, both for 2 degrees of wheel slip.

	VLE	VNLE
Observation weight	$R_{vle} = \begin{bmatrix} 3 \times 10^{-2} & 0 \\ 0 & 5 \times 10^{-2} \end{bmatrix}$	$R_{vnle} = \begin{bmatrix} 2 \times 10^{-3} & 0 & 0 \\ 0 & 2 \times 10^{-1} & 0 \\ 0 & 0 & 10^{-2} \end{bmatrix}$
State weight	$Q_{vle} = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-4} \end{bmatrix}$	$Q_{vnle} = \begin{bmatrix} 10^{-3} & 0 & 0 \\ 0 & 10^{-2} & 0 \\ 0 & 0 & 10^{-1} \end{bmatrix}$
Covariance error	$P_{vle} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$	$P_{vnle} = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}$

Table 4.7: Parameters for the computation of Kalman gains for both the VLE and the VNLE.

(section 3.1). The transition matrix for this estimator is time-varying and relies on the longitudinal velocity, and that implies the gain is also time-varying and only dependent of v_x. For this reason, the gain is computed offline, and a lookup table generated dependent of the longitudinal velocity, in order to keep the Kalman gain as close as possible to the optimal gain. The gain evolution is presented in Fig.4.11, which is computed using the observation and state weight matrices presented n Tab.4.7.

On the VNLE the same can not be done to the gain, since it is computed with the Jacobian matrix F_{VNLE} that relies on the three states used, implying that the gain must be computed online. The observation and state weight matrices for these computations are depicted in Tab.4.7.

The input on the VLE is the steering angle corrupted with noise as presented in section 2.6.4. The VNLE besides the steering angle uses also the longitudinal force F_x, that for simplification is used as a direct input from the vehicle. The observations of both estimators are the same, the yaw rate from the gyroscope corrected with the estimated bias from the ACF, and both velocity components from the PCF. With a small difference, where in the VLE the observation is $\beta_{PCF}=\text{atan}^{-1}(v_y/v_x)$, and also the longitudinal velocity enters directly in the transition matrix. In the VNLE, both velocities are direct observations.

The results of the sideslip estimate of both estimators, VLE and VNLE, can be seen in Fig.4.12, where they are set side by side with the real sideslip computed from the vehicle model before the sensors. In order to keep the graph clean, a rule was implemented that for longitudinal velocities below 3m/s, no sideslip exists. This rule is only applied in the first 5 seconds where the car is almost stopped. The problem was the noise, even small, if v_x=0.1m/s and v_y=0.1m/s, then the sideslip is β=45º, which resulted in high noise at the beginning.

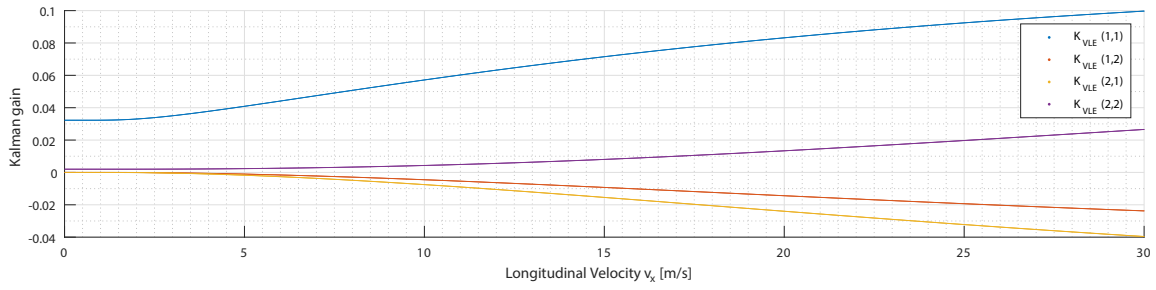

Figure 4.11: Evolution of Kalman gain entries along the longitudinal velocity, for the Vehicle Linear Estimator.

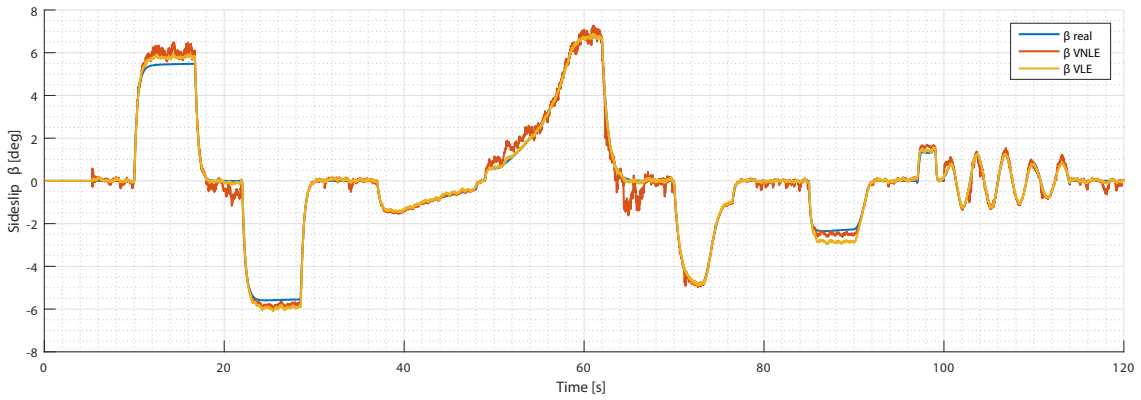

Figure 4.12: Sideslip results for the VLE and the VNLE in comparison with the real value.

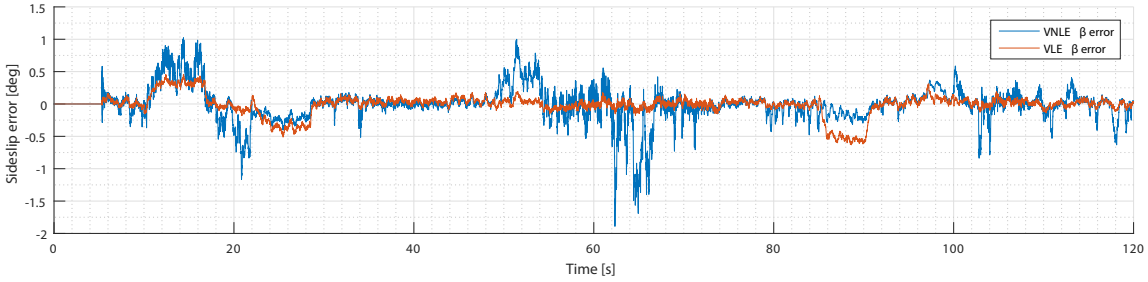

Figure 4.13: Sideslip error for the VLE and the VNLE in relation to the real value.

In Fig.4.12, both estimators can give the asked result, and close to reality. Is also evident that the VLE is better behaved than the VNLE, with a lower noise, and a smaller error. It should also be noted that in constant sideslip angle, for example, between [11, 17] seconds, a constant error exists. This is due to all the despised dynamics like wheel load change along the time, and the IMU rotation due to roll. In Fig.4.13 is presented the error to the real sideslip of both estimators, also in Tab.4.8 some statistical information is condensed, where the sideslip computed directly from the PCF velocities is also introduced. It is also clear by this table that the VLE is better than the two other options.

	max error	mean	σ	σ^2
PCF	9.8780	-0.0062	0.7202	0.5187
VLE	0.6311	-0.0119	0.1582	0.0250
VNLE	1.8907	-0.0095	0.2685	0.0721

Table 4.8: Comparison of statistical data for the computation of the sideslip angle for three different methods.

4.2.4 Unpredicted conditions

The above simulation uses the filters with perfect parameters relative to the vehicle, but this almost never happens in the real world, there are always some differences from non-modelled dynamics, or poorly calculated variables.

The following test aims to evaluate the performance of the filters in unpredicted conditions. The most common difference that can occur and the one that can drastically change the behaviour of the vehicle is the tyre-road interaction, especially when it is raining. Other common alteration is the road inclination or banking, that introduces an unbalance tyre load that can also influence the car behaviour.

For this simulation the filters were kept the same as before with regard to the vehicle, and filter weights. The changes were at the vehicle were the tyre model was change for the wet asphalt as seen in Fig.2.4, the road has now a banking with a -2% inclination in the Y direction. Besides all this, the references given of longitudinal velocity and yaw rate are the same as before, which are close to the car limit in dry conditions and almost impossible to achieve in wet. In Fig.4.14 is possible to see the new vehicle trajectory which is a deformed version of Fig.4.2.

In Fig.4.15 is showed the load of each tyre, for a road with and without the -2% inclination, as well as the vertical load assumed in the filters.

Since both the ACF and the PCF are kinematic filters, they are not affected by the tyres or the banking, so only the results of the VLE and VNLE are presented.

In Fig.4.16 is possible to see the results of both tested vehicle estimators. Like in the previous simulation (Fig.4.12), the VLE is a much cleaner signal than the VNLE, but in this case the VLE can not follow the real sideslip value by far. This problem is associated to the cornering stiffness approximation, where the one used (Tab.4.6) is for wheel slip angles below or very close to 2º, and in this case where the car is pushed to the limit, these angles are much higher as seen in Fig.4.17.

For example, using the wheel slip angle of the front left wheel on second 25, α_{FL}=21.9º, with a

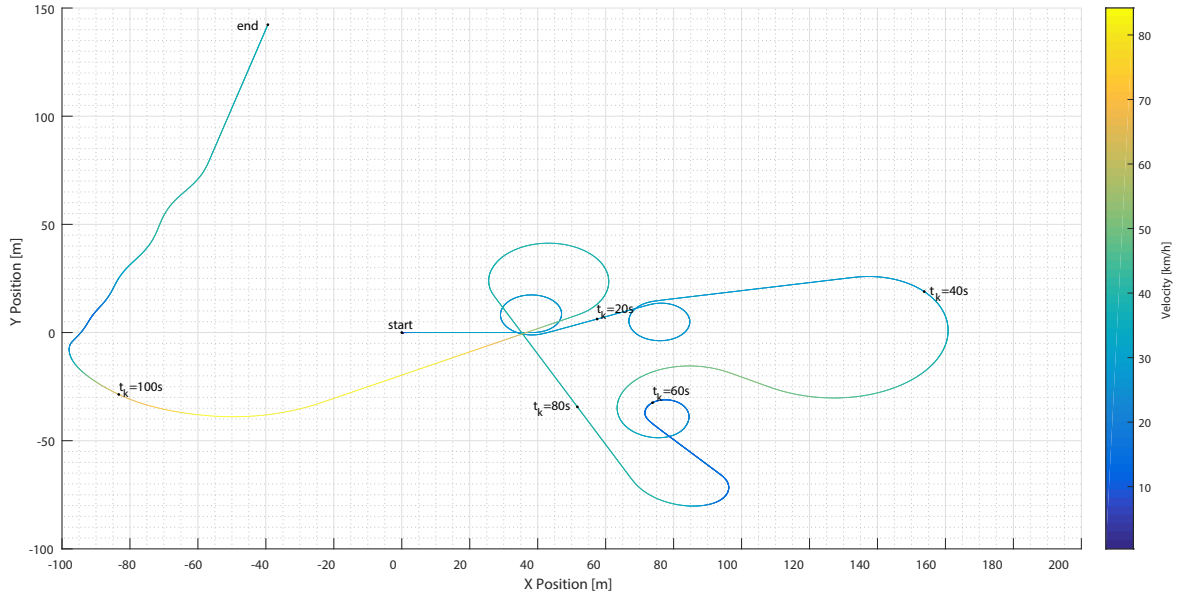

Figure 4.14: Trajectory of the vehicle for the new assumptions.

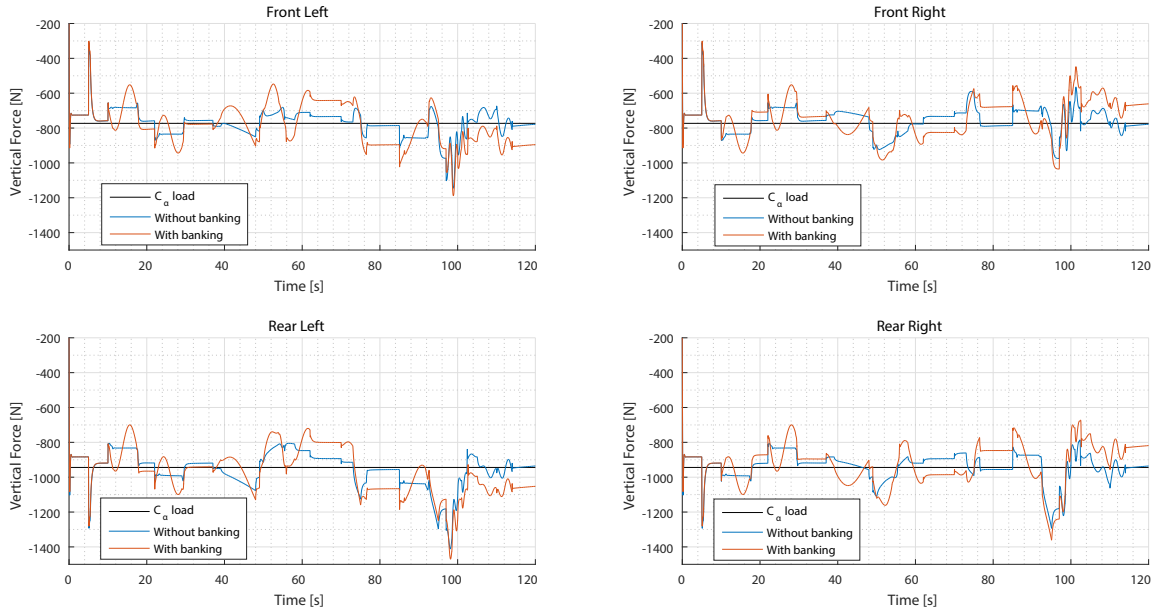

Figure 4.15: Vertical force in each tyre for comparison between a road with and without banking. The vertical load assumption for the cornering stiffness approximation is present for reference.

	Real	VLE		VNLE
		$C_\alpha\|_{\alpha=2°}$	$C_\alpha\|_{\alpha=5°}$	
Force [N]	533.78	12128.3	9081.1	1117.6
Diff.	-	$\times$22.7	$\times$17.0	$\times$2.1

Table 4.9: Comparison of lateral force estimated for α_{FL}=21.9º and Fz=744N using different methods, and differences between them.

vertical load of 744N, the difference in lateral force between models can be seen in Tab.4.9. Where at that moment the wheel is only generating 533.78N, where the VLE assumes it is doing 12128.3N, almost 23 times more than the real. This value is also compared with a cornering stiffness computed with angles up to 5 degrees, where it still expects values 17 times greater than the real one. On the other hand, since the VNLE uses the tyre equation, even a wrong one, it limits the maximum force, only expecting 2 times more, which is enough for the filter to compensate and give a close enough estimate.

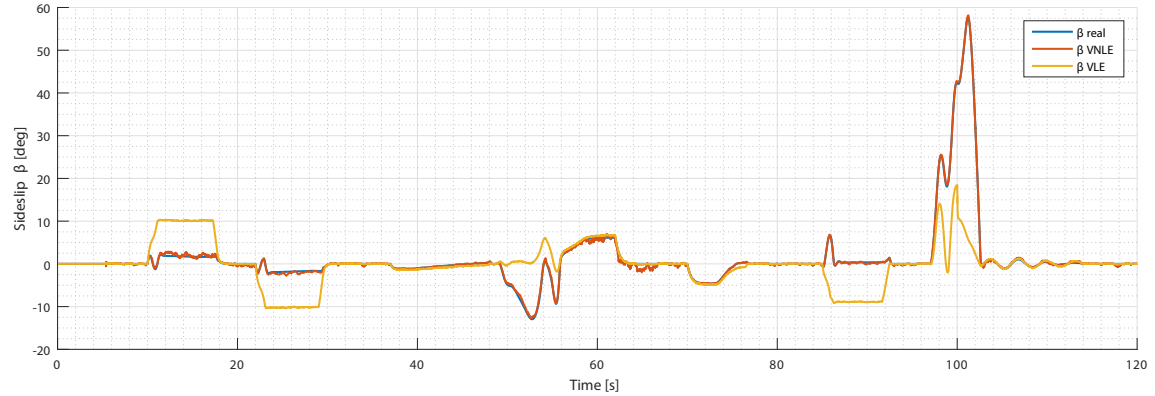

Figure 4.16: Sideslip estimate result for the VLE and VNLE, compared with the real sideslip, for the new unpredicted conditions of the vehicle.

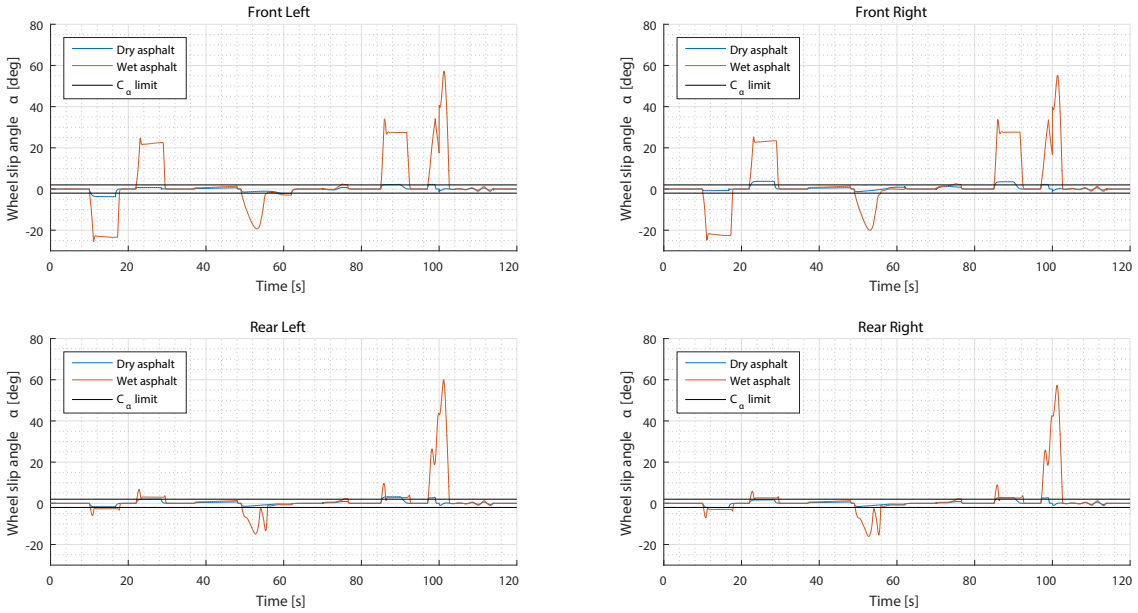

Figure 4.17: Wheel lateral slip angle for each tyre comparison for wet and dry road conditions. Also presented the cornering stiffness limits assumed.

In the end, it is possible to say that both estimators, the VLE and VNLE can be used, and the choice between them depend and the platform and test conditions used. If the vehicle is well known, and it is expected to have a behaviour inside the limits of what was defined for the estimator, then the VLE is recommended since it has a less noisy output, and is computationally lighter. On the other hand, if the vehicle and test condition are outside of the estimator limitation, then the VNLE is recommended, with the con of being computationally heavier and having a noisier output signal.

Chapter 5

Laboratory Implementation

This chapter presents the implementation of the estimator architecture in a micro-controller on a RC vehicle inside a laboratory. The hardware used is the continuation of the work done in [20].

In a first part it will be presented the hardware used, where the vehicle and the sensors will be explored. Since in this implementation the micro-controller will be responsible for managing all the communications and computations, a section about the software will depict the information flow. Lastly, the results obtained with this system, for the different filters, will be presented and compared with a secondary and more precise system for validation.

Since in the used vehicle was not possible to access the torque produced by the motor, the VNLE was not implemented and will not be referenced in this chapter.

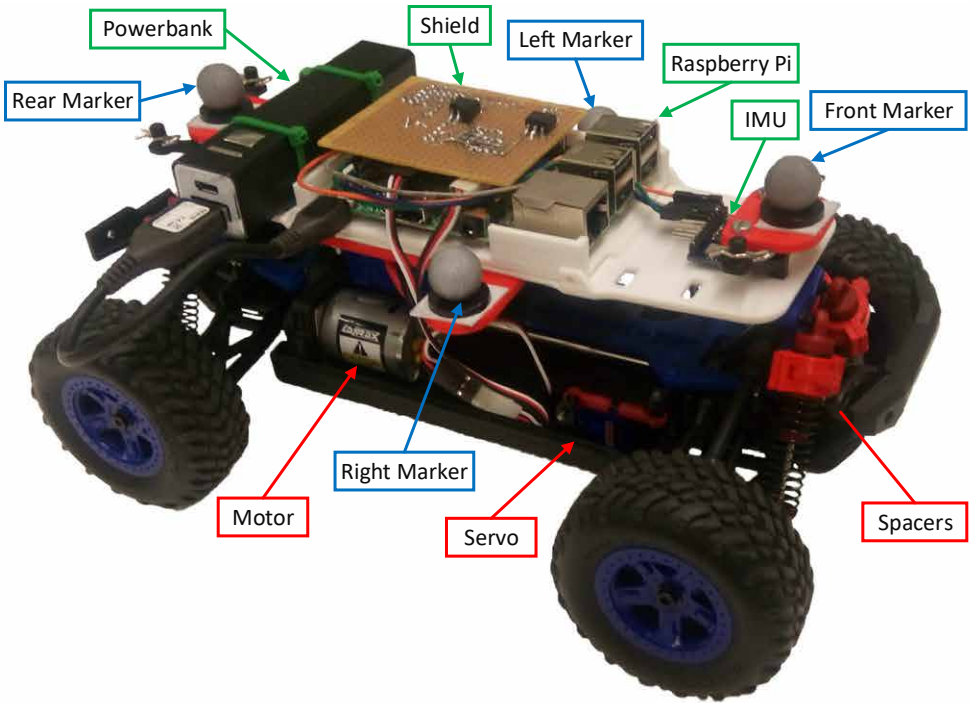

Figure 5.1: RC vehicle used in the laboratory, equipped with a Raspberry Pi 3, an IMU, and the infra-red markers. Several important components are identified. The battery is on the other side of the vehicle and not visible on the image.

5.1 Hardware

The vehicle used is presented in Fig.5.1, and consists in a radio-controlled 1:18 four-wheel drive car, to which the receptor and original remote controller were removed. The control of the car is then made by wireless and through the micro-controller.

The propulsion source is a single motor connected to the four wheels with two mechanical differentials, one per axle. The steering is only at the front wheels and is actuated by a small servo motor. Both actuators are supplied by a 7.2V NiMH battery with 1200mAh.

Due to the added weight, spacers had to be place in each shock absorber to increase the pre-load of the spring. This also helps to reduce the roll and pitch on the car.

The car was equipped with a standard Raspberry Pi 3 model B, which is the only micro-controller on board. This one is connected to an IMU through I^2C (Inter-Integrated Circuit), to two computers using the integrated wireless, and controlling both motors using two PWM (Pulse-Width Modulation) outputs. The Raspberry Pi 3 is supplied by a 2300mAh power bank.

To incorporate the raspberry and the remaining control and acquisition system, some parts were design and made with resource to 3D printing.

5.1.1 Sensors

The vehicle is equipped with two main sensors. The IMU (Fig.5.2) and a positioning system replacing the GPS that does not work inside the laboratory for obvious reasons.

The position is given by a system from Qualisys that consists in a set of six infrared cameras (Fig.5.3). These cameras acquire the reflection of the infrareds by the markers on the body to be identified. As seen in Fig.5.1, the vehicle has four marks. A dedicated software (Qualisys Track Manager) uses the information of the six cameras to triangulate the position of the body. The three position coordinates and three rotations are then transmitted over TCP/IP to any device listening at a rate of 100Hz.

This system sometimes shows a gap in the values due to bad data acquisition. The cameras are emitters and receptors of infrared light, and the floor of the laboratory is mirrored. In some locations it happens that the light from a camera is reflected to another one, and when the vehicle passes over this location, the system loses the tracking of the marker. In Fig.5.4 is possible to see this reflection by a

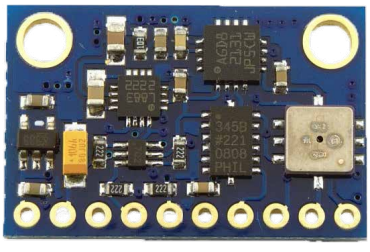

Figure 5.2: GY-80 IMU used in the vehicle for acquiring the accelerations, angular velocities and orientation.

Figure 5.3: Infra-red camera used by Qualisys software to acquire the vehicle position

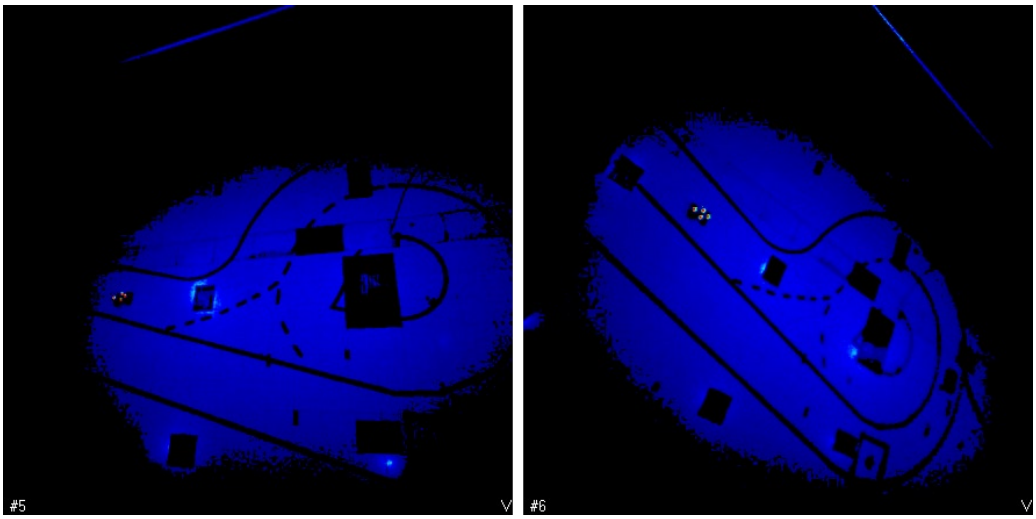

Figure 5.4: Infra-red intensity image of cameras five and six. The vehicle location is identified by four markers in each image, and the reflection problem is present by a lighter color.

brighter colour, an almost white glow. Some dark squares are placed in the floor to limit this effect.

The IMU is a GY-80, composed of a 3-axis accelerometer (ADXL345) from Analog Devices, a 3-axis gyroscope (L3G4200D) by ST Microelectronics, and a 3-axis magnetometer (MC5883L) by Honeywell. The accelerometer is configured for $\pm$4g's, the gyroscope for $\pm 2000^{o}$/s and the magnetometer for $\pm$1.3G (Gauss). The data from the three sensors is retrieved at 100Hz.

This IMU has several outliers on the gyroscope and on the X and Y axis of the accelerometer as seen in Fig.5.5. This is not a common noise, but most likely a problem in the breakout or in the sensor itself, since the outlier represents always a full buffer (negative or positive). The hypothesis of being a problem with this specific IMU or an implementation error is also discarded since in [20] the same problem arises. To overcome this situation an one dimensional median filter with a window of three is implemented using the raw data before any conversion. The result can be seen in Fig.5.5(c) for the accelerometer and Fig.5.5(d) for the gyroscope.

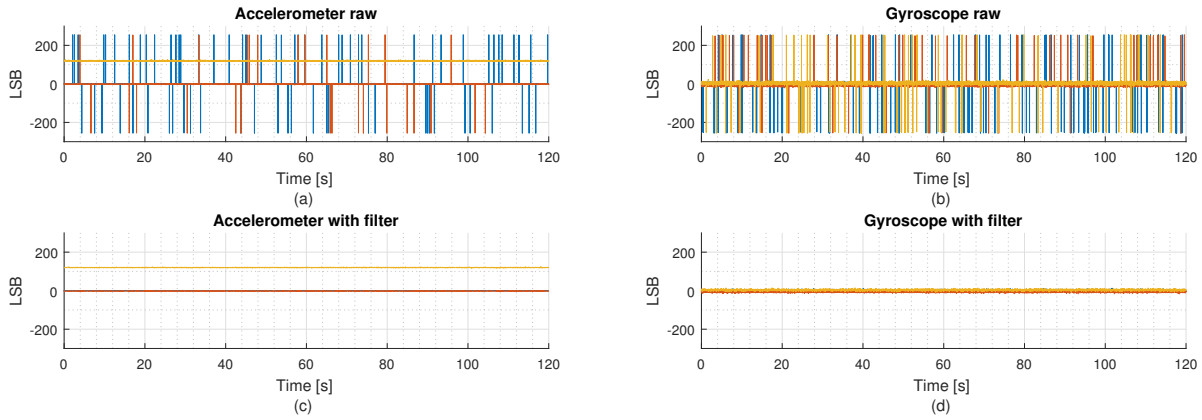

Figure 5.5: Comparison of the raw data of the accelerometer (a) and gyroscope (b), with the data using an one dimensional median filter with a window of three. All the data in LSB. The three axis are presented, x-axis in blue, y-axis in orange and z-axis in yellow.

	Accelerometer [mg]			Gyroscope [dps]			Magnetometer [mG]		
	x	y	z	x	y	z	x	y	z
Range		±4g			±2000dps			±1.3G	
Scale [*/LSb]	7.7	7.6	7.9	0.07	0.07	0.07	0.92	0.92	0.92
Offset [LSb]	0	-2	-5	0	0	0	n/a	n/a	n/a
Mean	-0.9	0.2	988.4	0.11	-0.42	0.25	n/a	n/a	n/a
σ	3.7	4.2	5.8	0.18	0.19	0.22	1.59	0.86	4.08
σ^2	13.8	17.4	33.8	0.03	0.04	0.05	2.53	0.74	16.65

Table 5.1: Information relative to the IMU sensors. Results for configurations, calibration and acquired data. Accelerometer data in milli-g, gyroscope in milli-degrees per second and magnetometer in milli-Gauss. Ranges equal for 3-axis. Where σ represents the standard deviation and σ^2 the variance.

A resume of the IMU sensor data can be found in Tab.5.1. The scales of the accelerometer and the offsets were determined by experimental data.

The mean and offset of the magnetometer are not presented since it has a special calibration described in Appendix B. In this particular implementation the magnetometer is not very useful due to the laboratory topology. As seen in Fig.5.6, underground power and communication cables supply different equipment's in the laboratory. Since the IMU is very close to the ground, its affected by the magnetic fields generated by the cables. This distortion can not be accounted in the calibration since it is a characteristic of the environment and goes against the assumptions of a constant magnetic field.

Besides these sensors is also necessary the steering angle for the VLE. Since no sensor present in the car can give this information, the value is acquired through the actuation signal of the servo. As it was identified in [20], the actuation is a PWM signal with a period of 10ms and a duty cycle comprehended between 10% and 20%, where 15% is the neutral. By means of experimental data, equation (5.1) was identified as the relation between the PWM duty cycle and the steering angle.

$$\delta_r[deg] = (\text{duty cycle[\%]} - 15)\,\frac{20}{25} \tag{5.1}$$

It is well known that some of the servo dynamic is being neglected, but the steering has some slack that introduces more uncertainty than the ignored dynamic.

Figure 5.6: Laboratory ground with identification of the cables location underground.

5.1.2 Wired Connections

The connections on the vehicle, as stated before, are based on the work of [20], with a few upgrades.

The general schematic can be seen in Fig.5.7. The Electronic Speed Controller (ESC) is controlled by a PWM signal as common servo motor, but instead of requiring power from the control connector, it is directly supplied by the vehicle battery. The servo that actuates the steering is also controlled by a PWM signal. The power to the servo comes from the ESC.

The major upgrade to the work in [20], is the addition of a shield to the Raspberry Pi 3, that makes the interface from the micro-controller to the IMU and actuators of the car. The Raspberry Pi is supplied by a power bank through an USB cable.

The purpose of the shield (Fig.5.8) is to facilitate the connections, and to protect the Raspberry Pi from wrong connections. It also reduces the problems of wires disconnections due to vibrations. The documentation of the shield can be found in Appendix C.

This shield consists of two optocoupler, one for each PWM signal. One LED attached to a Raspberry Pi output for debug. A voltage regulator to retrieve power from the battery to the PWM. And a second LED indicating that the tractive system of the car in on. The board is connected to the header of the Raspberry PI and has only three pin arrays. One for the IMU, one for the servo, and one for the ESC.

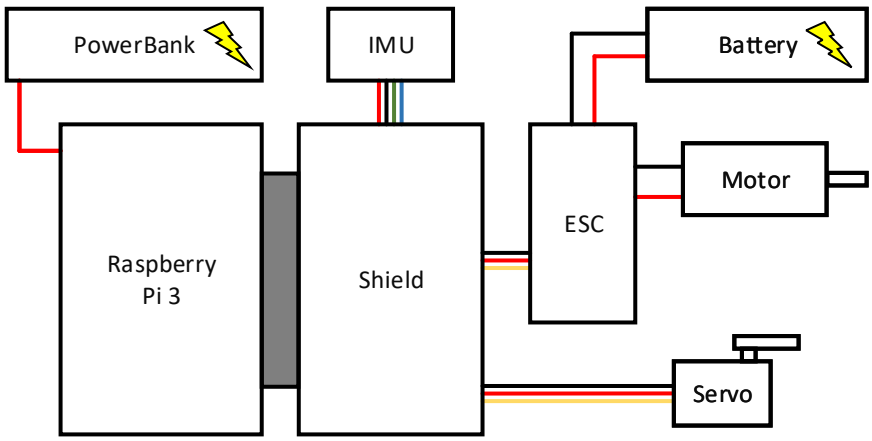

Figure 5.7: RC Vehicle wiring schematic.

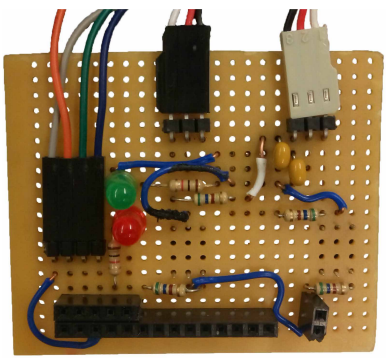

Figure 5.8: Shield that makes the interface between the Raspberry Pi 3 and the sensors and actuators.

5.2 Software

5.2.1 Wireless communications

In Fig.5.9 is depicted the wireless communication flow. The computer responsible for the Qualisys software, receives the information from the cameras (Cam) and sends one message with the three positions and three rotation components over TCP to the vehicle.

The second computer or laptop has two programs running in simultaneous. The first (Prgm1) is a program created by [20] and is connected by USB to a remote controller. The steering and velocity information acquired by the remote is then sent by UDP to the vehicle. The second program (Prgm2) is connected by TCP with the vehicle and is a two way communication channel. It sends start and stop information to the program on-board the vehicle and receives results from the same.

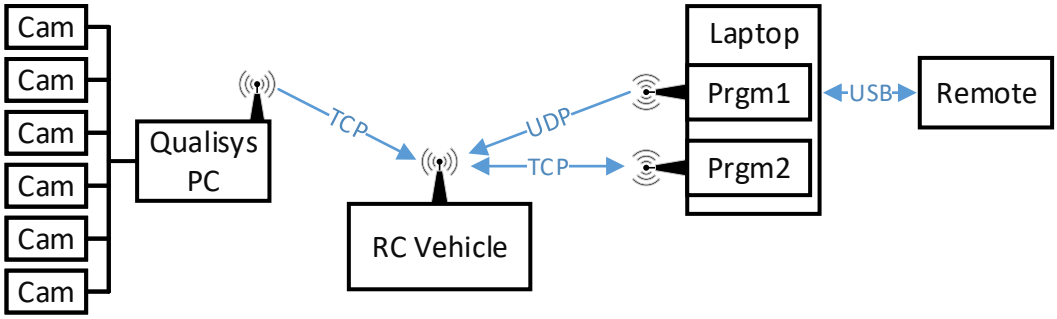

Figure 5.9: Diagram of wireless communication flow.

5.2.2 Program on-board the vehicle

A program was created to run in real-time inside the Raspberry Pi 3. This one was written in C code, and besides the common Linux base libraries like ”math.h”, it only relies more on the ”wiringPi.h” [44] for the GPIO input/output interface. All the other necessary libraries like for the Kalman filter were self-developed.

The general structure and pseudo-code algorithm can be seen in Fig.5.10. The program is divided in a main program and three parallel threads.

The first thread is for the IMU. This one, starts by making the configuration of the sensors registries, and to test the acquisition and speed of the communication. After that, enters in an infinite while loop. In this loop the thread acquires the data from the three sensors, converts the raw data to usable information with all the calibrations, sends the data to the buffer and writes the information in a file. A timer function ensures that the cycle only restarts at the desired time step. The thread only exits when it gets the information from the main.

The second is for the remote control. This thread starts the PWM signals in the neutral value (15%), and starts an UDP channel with the program on the laptop responsible for the remote. It enters a while loop where it waits for a new message from the laptop, converts the message to duty cycle and updates the output PWM values. These values are also sent to a buffer accessible by the main function. When

the main breaks the while, the thread puts the PWM signals to neutral before it ends. This function uses the hardware PWM capabilities of the Raspberry Pi 3, to remove load from the CPU.

The third thread takes care of the communication with the Qualisys computer. It starts by sending the configuration messages necessary to the Qualisys. The while loop waits for a new message and converts the raw data to usable values. These values are sent to the buffer, and written on a log file. The while cycle is just stopped by the main function.

The main function, initiates every necessary variable, and starts the three threads. Before advancing, it waits for valid messages from every thread. After that, it initializes the three filters, and waits for the order from the laptop to start the cycle. The cycle starts by retrieving all the data available from the buffers, and then runs the filters. The results are sent over TCP to the laptop, and logged in a file. The while cycle is only stopped with information from the laptop. A function ensures that the defined time

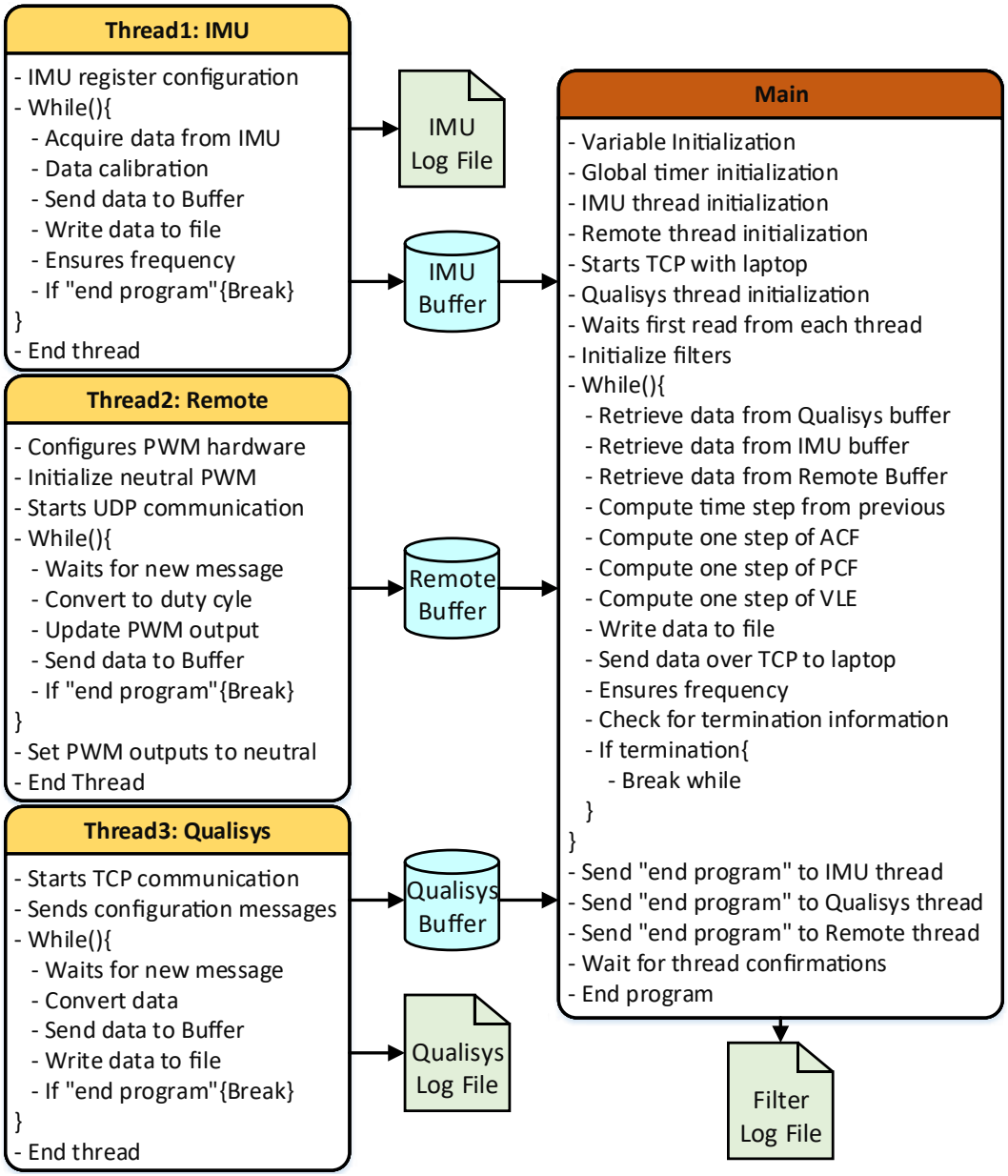

Figure 5.10: Program pseudo-code algorithm with interaction between threads.

step is satisfied.

Every buffer is for a single message only. If the main did not read the message in time, then the content of the buffer is rewritten. Three log files are used to ensures every function can make the log at their own pace, however every message written has a time stamp relative to the global timer defined in the main.

5.3 Tests and Results

Next the results for the different implemented filters on the vehicle will be presented. Since in this platform all sensors could be acquired at 100Hz, the estimates are done at the same frequency.

For the validation of the estimated results, is used the data from the Qualisys. The heading is a direct output of the system, however the same does not happen with the velocities. These ones are computed by a simplistic numerical differentiation of the position. As seen in Fig.5.9, all the data from Qualisys passes only through the vehicle, and the sole reason is to get the same time stamp as the estimates.

The results will be presented for the different filters and compared whenever possible with the Qualisys data. As its going to be seen, and as it was already mentioned, the limitation is the magnetometer.

5.3.1 Attitude Complementary Filter

As already stated, the magnetometer its almost useless in the laboratory environment. Even so, to maintain the consistence with the remaining work, the magnetometer is used with a small modification to the ACF.

The ACF in this implementation uses another functionality of the Kalman Filter, the integration of two different readings for the same state. Instead of only using the yaw from the magnetometer, it is also used the yaw from the Qualisys. The objective is to still have the uncertainties of the magnetometer, but a little close to reality to be useful. The observability matrix of the ACF is then given by (5.2) and the Kalman gain is a 2×2 matrix. The weights and gains used for the ACF are presented in Tab.5.2. The first entry of the R_{acf} correspond to the data from the magnetometer and the second one to the Qualisys. As can be seen, it was still given more importance to the angle from the magnetometer.

$$C_k = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \tag{5.2}$$

In Fig.5.11 is presented the results of the yaw estimate using the ACF. Also, the Qualisys and magnetometer angles are present and as can be seen, the magnetometer yaw has a significant difference

Observation weight	State weight	Filter Gain
$R_{acf} = \begin{bmatrix} 10^3 & 0 \\ 0 & 7 \times 10^2 \end{bmatrix}$	$Q_{acf} = \begin{bmatrix} 10^{-1} & 0 \\ 0 & 10^{-1} \end{bmatrix}$	$K_{acf} = \begin{bmatrix} 9.58 \times 10^{-3} & 1.37 \times 10^{-2} \\ -6.34 \times 10^{-3} & -9.06 \times 10^{-3} \end{bmatrix}$

Table 5.2: Parameters for the Attitude Complementary Filter used in the RC vehicle.

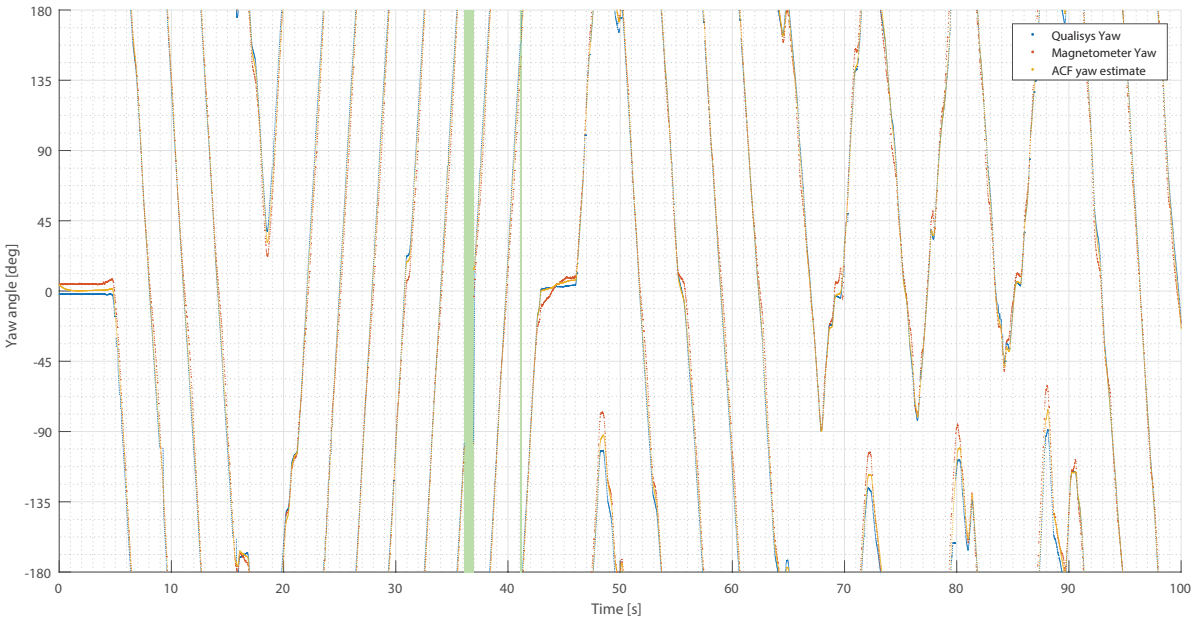

Figure 5.11: Yaw angle results comparison between the true Qualisys angle, the yaw computed from the magnetometer and the estimated value from the ACF.

to the Qualisys angle. The ACF estimation is always somewhere between the two readings, and always a little closer to the magnetometer as specified with the weights.

The Qualisys data has some gaps that are going to be explored ahead. The ACF using the two sources of yaw angle and the yaw rate from the gyroscope, can give an estimate without gaps of information.

Also in Fig.5.11 two light green zones are represented. These are zones where the Raspberry Pi has blocked for unknown reasons, and all the data stopped for some time. This freeze is visible in all the data presented.

In Fig.5.12 is presented the yaw rate bias estimated by the ACF for the Z-axis of the gyroscope. Also, the mean offset presented in Tab.5.1 is represented. The initial spike is due to the initial yaw adjust seen in Fig.5.11. The estimate, as seen, stays around the expected offset and the variations correspond to the roll influence of the vehicle.

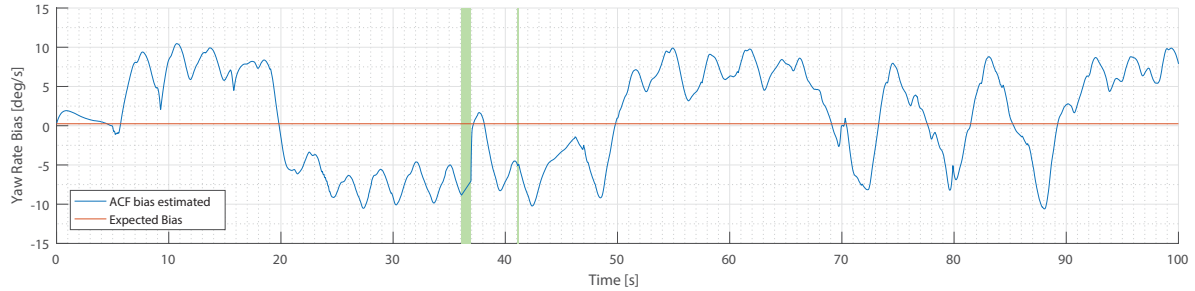

Figure 5.12: Yaw rate bias estimate from the ACF compared with the expected static bias for gyroscope.

5.3.2 Position Complementary Filter

The PCF did not suffer any modifications as the ACF above. The position is given by the Qualisys, and the accelerations from the accelerometer with the corrections associated with model (2.34). The PCF is a time-varying filter that depends on the yaw angle estimate from the ACF. The yaw angle as seen before will be the major source of error.

The weights and gains for this filter are presented in Tab.5.3. These gains were adjusted to this particular sensor, specially the position, which is more precise than the one given by a GPS.

Observation weight	State weight	Filter Gain
$R_{pcf} = \begin{bmatrix} 10^{-4} & 0 \\ 0 & 10^{-4} \end{bmatrix}$	$Q_{pcf} = \begin{bmatrix} 0.1\mathbf{I}_2 & 0 & 0 \\ 0 & 10\mathbf{I}_2 & 0 \\ 0 & 0 & 10^{-5}\,\mathbf{I}_2 \end{bmatrix}$	$K_{pcf} = \begin{bmatrix} 0.757\,\mathbf{I}_2 & 0 & 0 \\ 0 & 6.98\mathbf{I}_2 & 0 \\ 0 & 0 & -0.068\mathbf{I}_2 \end{bmatrix}$

Table 5.3: Parameters for the Position Complementary Filter used in the RC vehicle.

The results for the velocity components is presented in Fig.5.13 for the longitudinal velocity and in Fig.5.14 for the lateral component.

Before analysing the PCF results, the Qualisys velocities deserve a closer look. In both components the velocity components of the Qualisys have outliers all around. This is a sum of two problems. The first, as stated before, is the computation. The Qualisys does not provide the velocity in real time, so it must be computed using the position. This numerical differentiation often creates these situations. The second problem is related with the receiving data. For some unknown reason, sometimes the buffer receiving the TCP data stacks several messages. When the thread retrieves the information, it gives a

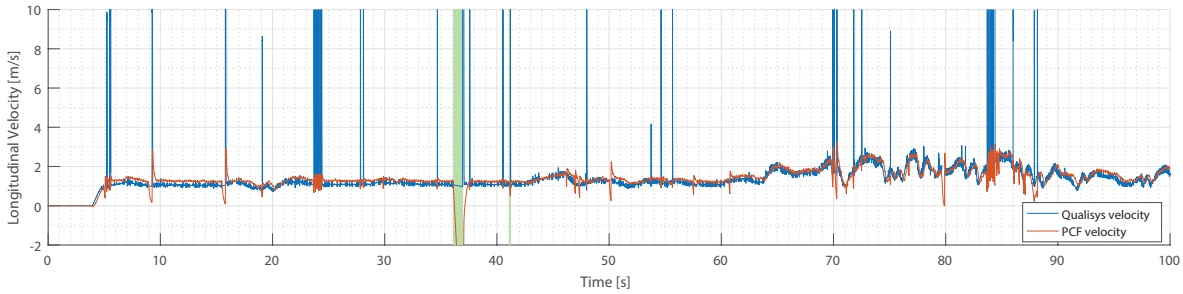

Figure 5.13: Longitudinal velocity component in the vehicle frame. Comparison between the differentiation from the Qualisys position, and the estimate from the PCF.

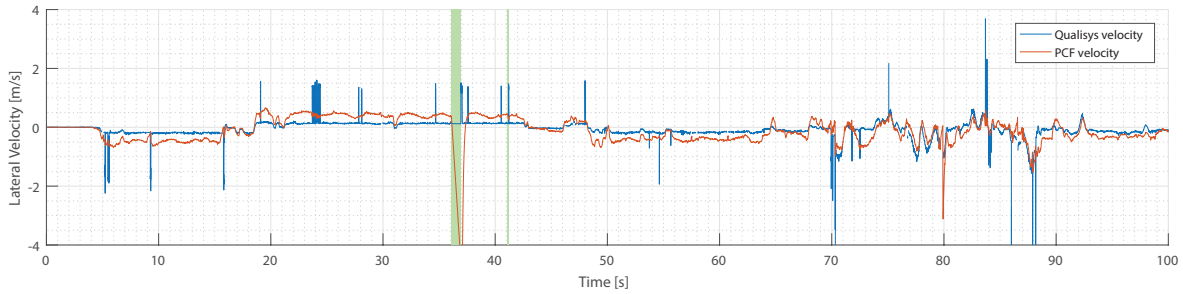

Figure 5.14: Lateral velocity component in the vehicle frame. Comparison between the differentiation from the Qualisys position, and the estimate from the PCF.

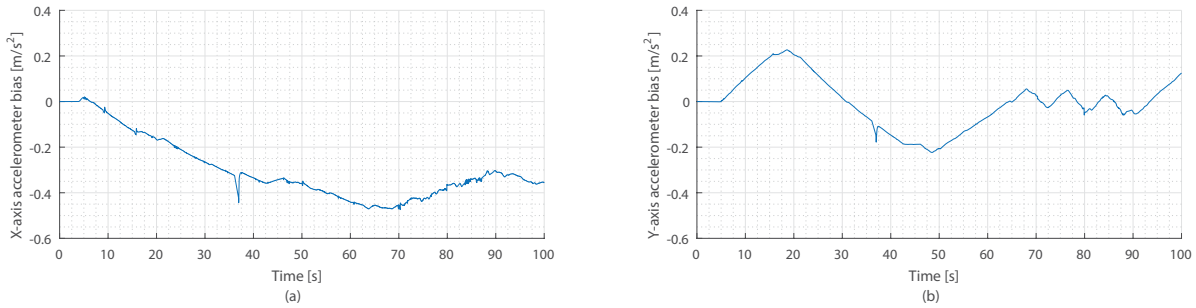

Figure 5.15: Accelerometer bias estimate from the PCF. In (a) the x-axis component, and in (b) the y-axis component.

time stamp. If several messages are stacked, and computed almost instantaneously, the time step will not be the true 0.01sec, but less, which originates a spike in the velocity.

Another situation that occurs is when the vehicle passes in zones were the cameras cannot fix the car position, as seen before. At these moments no message is sent and a gap appear in the data.

For the estimates and filters these velocities are not used, so the only problem is in the presentation of the results. However, the error in the time stamp will be noticed along the results even if its influence is minimal.

In Fig.5.13 is presented the longitudinal velocity component. In general, the results present a small deviation from Qualisys. This is associated to the error in the yaw angle. A similar deviation is also seen in Fig.5.14 for the lateral component.

Also, some spikes appear along the estimate, some due to the time stamp error in the position like at 16sec, and others are due to missing data also from the Qualisys like at the 50sec.

The analysis of the lateral velocity in Fig.5.14 is similar to the longitudinal one. With the only difference that the deviation is more pronounced due to the range of values.

In Fig.5.15 is also presented the bias estimate of the accelerometer for both components. And for both axis is clear that they try to converge for a value. In the x-axis is around the -0.4m/s^2, and for the y-axis around the 0m/s^2. Both, as expected respond in order to compensate the rotations of the vehicle.

5.3.3 Vehicle Linear Estimator

For using the VLE, a problem arises. The VLE depend on the cornering stiffness of the interaction between the tyres and the road surface. However, for the RC vehicle and the surface of the laboratory ground there is no data. To implement the VLE, first is presented the model identification, and after that is presented the implementation and results.

Model identification

The approach taken consists in using the Matlab System Identification Toolbox, to adjust acquired data from the RC vehicle to a modified version of model (2.26). The new model (5.3) as seen in [26] uses as state variables the sideslip of the vehicle and the yaw rate.

$$\begin{bmatrix} \dot{\beta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} -\frac{C_{\alpha f}+C_{\alpha r}}{mv_x} & -\frac{aC_{\alpha f}+bC_{\alpha r}}{mv_x^2}-1 \\ -\frac{aC_{\alpha f}-bC_{\alpha r}}{I_\psi} & \frac{-a^2C_{\alpha f}+b^2C_{\alpha r}}{I_\psi v_x} \end{bmatrix} \begin{bmatrix} \beta \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \frac{C_{\alpha f}}{mv_x} \\ \frac{aC_{\alpha f}}{I_\psi} \end{bmatrix} \delta \tag{5.3}$$

The data used for the identification correspond to a set where the car was at an almost constant velocity around 0.77m/s. From the fit resulted the state transition and input matrices (5.4), with a percentage of fit to estimation data of 93.9% for state one and 87.9% to state two.

$$A_k^{vle} = \begin{bmatrix} -4.473 & -0.114 \\ -0.070 & -8.633 \end{bmatrix} \quad , \qquad B_k^{vle} = \begin{bmatrix} 1.174 \\ 22.87 \end{bmatrix} \tag{5.4}$$

The fit results are for a single longitudinal velocity. To expand the results to a time-varying model a direct relation was made with (5.3) resulting in (5.5) which is time-varying and dependent of the longitudinal velocity.

$$\begin{bmatrix} \dot{\beta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{3.4576}{v_x} & \frac{0.5298}{v_x^2}-1 \\ -0.07032 & \frac{-6.6733}{v_x} \end{bmatrix} \begin{bmatrix} \beta \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \frac{1.371}{v_x} \\ 22.87 \end{bmatrix} \delta \tag{5.5}$$

VLE implementation and results

This model is still continuous and is only converted to discrete inside the micro-controller. The conversion is made with resource to the already presented property (3.37). The matrix exponential is computed using (5.6), where I is the identity matrix.

$$e^A = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots + \frac{A^n}{n!} \tag{5.6}$$

The stop criteria implemented for the convergence of (5.6) is $n = 200$ or the maximum difference between two consecutive sum terms is less than 10^{-9}.

With the model converted to discrete, the Kalman Filter can then be implemented in the micro-controller as the others. In this one, since the filter is time-varying, the Kalman Gain is computed iteratively inside the Raspberry Pi, for each time step. The weights are presented in Tab.5.4.

In Fig.5.16 is presented the estimation result of sideslip angle from the VLE and compared with the one from Qualisys.

As seen before, the Qualisys data sometimes has gaps and missing information from passing in blind zones for the cameras. These sources of error appeared before in the yaw angle and in the velocities, and the sideslip is no exception. These errors originate the spikes on the Qualisys data.

Observation weight	State weight	Covariance error
$R_{vle} = \begin{bmatrix} 10^{-2} & 0 \\ 0 & 0.5 \times 10^{-2} \end{bmatrix}$	$Q_{vle} = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{bmatrix}$	$P_{vle} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Table 5.4: Parameters for the Vehicle Linear Estimator used in the RC vehicle.

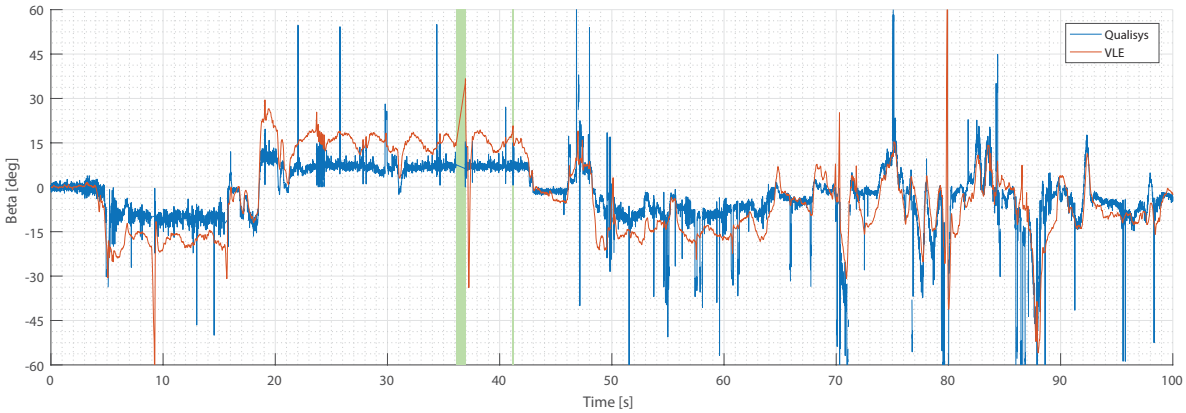

Figure 5.16: Sideslip result from the VLE compared with the sideslip angle from the Qualisys.

Analysing now the VLE estimate result. As seen before in the PCF the velocities components have some deviation from the Qualisys data, and the same is verified in the sideslip. Even though, with the VLE the difference is significantly lower, it still appears and generates a considerable error. This deviation is more evident in the constant velocity turns between [5, 16]sec and between [19, 42]sec. However, in faster turns as seen from 74sec till the end, the error appears to be smaller.

Also in the constant velocity turns an oscillation appears in the estimate result. This is the influence of the magnetometer yaw angle. When the vehicle passes closer to the underground cables (Fig.5.6) the magnetic field changes for the magnetometer giving a considerable error in the sideslip estimate.

5.3.4 Results discussion

The final result of the sideslip angle presented in Fig.5.16 has some significant error to the Qualisys angle. This was attributed mainly to the magnetometer yaw error but other factors also contribute to this. The model used is a linear one since the VNLE could not be implemented. Also, the linear model was achieved through identification of real data is for a single velocity and extrapolated for the remaining ones. With these, the weights used for the VLE, had to be wide enough to contemplate all the uncertainties sources.

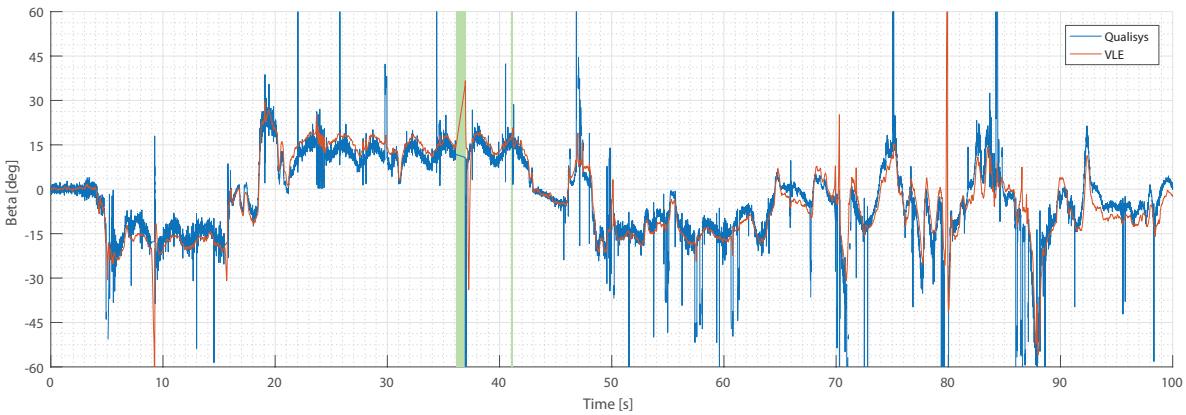

Figure 5.17: Sideslip result from the VLE compared with the sideslip angle from the Qualisys, with the assumption of perfect yaw angle.

	mean	σ	σ^2
VLE (original)	-0.4695	10.6579	113.5108
VLE (ACF yaw)	-0.8077	9.6955	94.0026

Table 5.5: Comparison of statistical data for the computation of the sideslip angle for the result with the original VLE and with the yaw hypothesis proposed.

To corroborate the wrong influence introduced by the magnetometer a second test is made. The data used is the same as before, as well as the weights and gains used in the filters. The only assumption is that the yaw angle used for computing the velocity components from the Qualisys is the yaw angle estimated by the ACF. In resume, is assuming that the yaw angle from the ACF is the correct one.

In Fig.5.17 is presented the comparison of both sideslip angles assuming the same yaw angle. And as can easily be seen by comparison with the results in Fig.5.16, the error is much smaller, and all the dynamics are caught by the estimator. Additionally, the statistical results of the error for both cases can be seen in Tab.5.5. The remaining deviations can then be associated with the linear model used, instead of a non-linear one.

Chapter 6

Formula Student Implementation

In this chapter is presented the implementation of the estimator architecture in a real Formula Student prototype. Unfortunately, at the time available for the tests with the car, the estimators were not yet integrated in a platform capable of interacting directly and in real time in the vehicle. To overcome this situation, all the data from the car was logged and the filters used in offline.

For the test, the car was the already mentioned FST06e (Fig.4.1), equipped with two different and completely independent systems (Fig.6.1). The first is the embedded acquisition line of the car, which is an integral part of the car, and its control systems. The second is a DGPS (Differential Global Positioning System), directly capable of providing the heading of the car as well as the velocity components, elements to make the validation of the vehicle sideslip estimate.

Throughout the chapter both systems will be closely explored, as well as the test conditions, and adjusts made to the acquired data. The estimate of each filter will be put side by side with the DGPS data, and the results discussed.

It should be noted that, the DGPS data is only used to compare results, and its data is not used, at any time in the estimators.

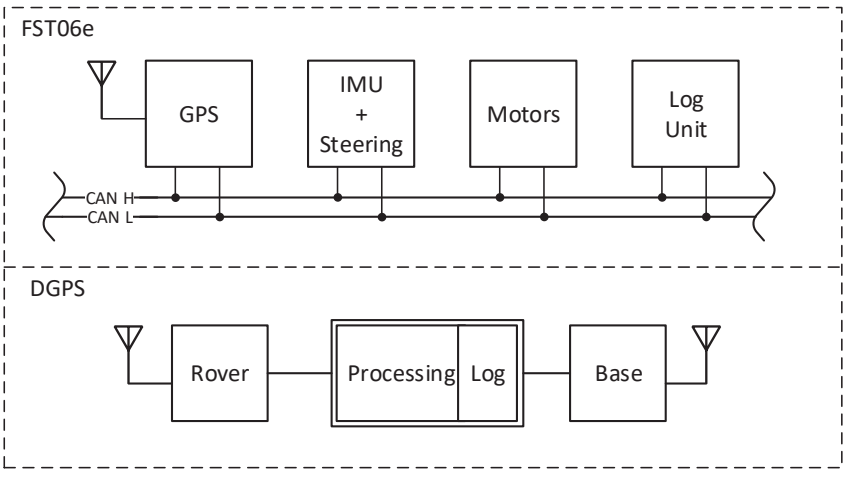

Figure 6.1: Acquisition architecture used in the vehicle. The FST06e represent the systems already implemented in the car, and the DGPS is the equipment used for the validation. The ”CAN H” and ”CAN L” suggest the CAN high and low lines of a CAN-bus protocol.

6.1 Test Platform

As already stated before, the vehicle used was the FST06e. The details of the car will not be mention in this chapter since they were already explored before. For this test run the car was stripped of all aerodynamic devices (undertray, front and rear wings) as seen in Fig.6.2. Two main reasons motivated this decision. The first was to remove the influence of the aerodynamic forces acting on the car. The second and main reason was for the DGPS antennas to have a cleaner view of the sky, and to have fixing points for the supports. Also having the antennas at roughly the same height was desirable to remove any unwanted influences.

The test was made in a university campus parking lot, with roughly 25x60 meters of useful asphalt as seen in Fig.6.3. Since the track was short, high velocities could not be reached in safety, nevertheless some peak velocities around 50km/h and an average of 30km/h was still achieved. The ground is not perfectly flat, with a convex deformation on the larger side (hill top), and a small slop on the other.

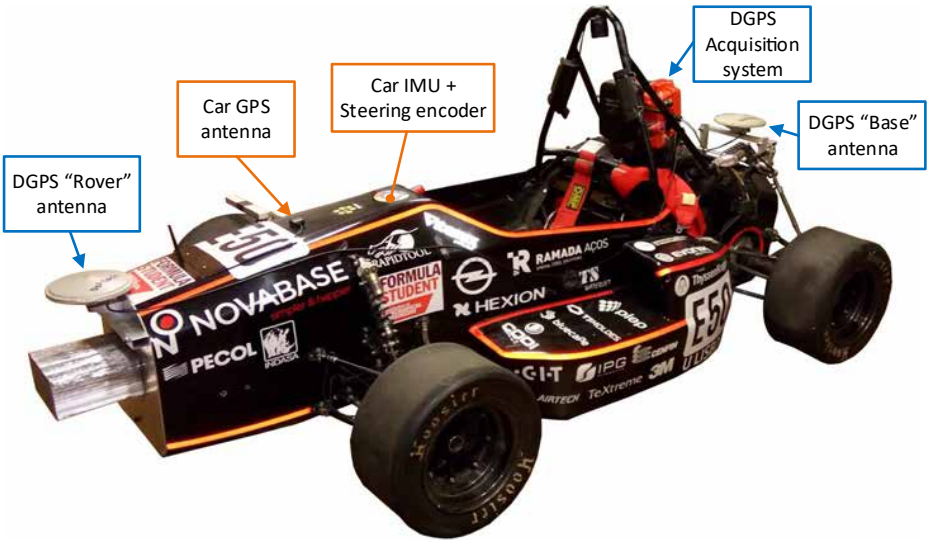

Figure 6.2: FST06e Acquisition system with DGPS. in orange are the sensors belonging to the car, and in blue the elements of the DGPS.

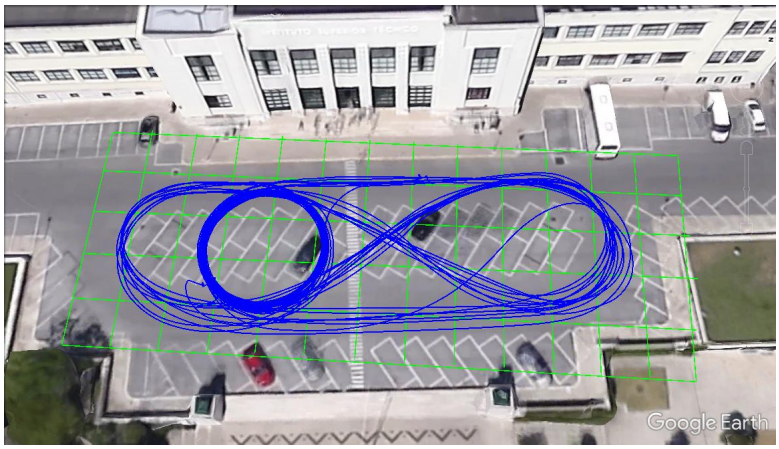

Figure 6.3: Trajectory of the car (blue) overlapped with a satellite image using Google Earth. A grid (green) with 5 meters division is also present for better notion of space.

6.1.1 Vehicle Acquisition system

The FST06e relies on a modular electronic circuit for all the monitoring, controls and acquisition. This circuit spreads all along the car, and consists in several modules connected using a CAN-bus (Controller Area Network) line working at 1Mbit/s. Even though the car has several modules, for the information required during this test, only four are directly required (Fig.6.1).

The GPS module, which receives the computed position information from a SkyTraq S1216F8 GNSS receiver at 25Hz and sends it to the CAN-bus line.

The IMU + steering encoder, as the name says, acquires the data from the IMU as well as the steering encoder value. The IMU is a GY-80 10dof, that has a 3-axis accelerometer ADXL345 from Analog Devices configured for $\pm$4g's, a 3-axis gyroscope (L3G4200D) from ST configured to $\pm 500^{o}$/s, and a 3-axis compass from Honeywell configured to $\pm$1.3Ga. The steering encoder consists in a rotational potentiometer connected to the steering column. Both sensors are acquired at 100Hz.

The third module is connected to the motors. The motors used in the car are from Siemens and have a dedicated controller by the same manufacture. In order to actuate these motors a module (the motors module) makes the bridge between the vehicle CAN-bus line and the motors controller. Also, these controllers return some values like motor speed, power, and the torque needed for the EKF. This module returns data at 20Hz. The torque at each motor is computed using the voltage and current to achieve the electric power, and with the motor speed by the encoder. These computations are all internal to the motor controller.

The fourth and final module is a log unit. This module receives every message in the CAN-bus line and writes it down in a SD card.

Each of these modules uses a generic board (Fig.6.4) to compute the necessary data which consists of a dsPIC30f4013 from Microchip working at 30MHz. Every CAN-bus message has a time stamp associated.

6.1.2 DGPS Acquisition system

The DGPS acquisition system (Fig.6.5) uses two ASH-660 GNSS antennas from Ashtech connected to a dedicated Ashtech MB100 board that besides the position and velocity also returns the heading angle. The data returned from the board is pre-processed and logged (orange briefcase). The system

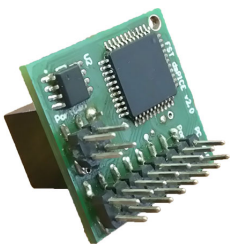

Figure 6.4: Board used in FST06e modules with the dsPIC30f4013.

Figure 6.5: DGPS acquisition system. Two antennas and the logging and supply system.

has its own power supply, and the data can be retrieved by connecting a computer to the acquisition system. Unfortunately, this system was only working at 10Hz. This DGPS uses a ”rover-base” system. The ”base” antenna is used as reference, and the position and velocity are given for the ”rover” location.

6.2 Data treatment

Unlike simulations, in real data acquisition, most of the time the data does not return in perfect usable conditions, and some processing must be done.

6.2.1 GPS conversion

This situation is particularly true for the GPS. In simulation is assumed that the position is given in meters, in reality the GPS returns Latitude, Longitude and Altitude (LLA) and ECEF (Earth-Centered, Earth-fixed) coordinates. To have useful data a transformation is made from the previous coordinate systems to ENU (East North Up) [45]. Is chosen a point or location of reference with latitude λ_r, longitude ν_r and an ECEF position $\{X_r,\, Y_r,\, Z_r\}$, that is going to be the origin of the new referential. The ECEF to ENU transformation consists in approximating a plane centred at that point. This is only valid for small distances, but given the size of the Earth, the size of the track used, or other type of motorsport track, this is still a valid approximation.

$$\begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix} = \begin{bmatrix} -\sin\lambda_r & \cos\lambda_r & 0 \\ -\sin\nu_r\cos\lambda_r & -sin\sin\lambda_r & \cos\nu_r \\ \cos\nu_r\cos\lambda_r & \cos\nu_r\sin\lambda_r & \sin\nu_r \end{bmatrix} \begin{bmatrix} X_p - X_r \\ Y_p - Y_r \\ Z_p - Z_r \end{bmatrix} \tag{6.1}$$

The transformation is given by (6.1), where $\{X_p,\, Y_p,\, Z_p\}$ is the ECEF coordinates to make the conversion to ENU, and $\{x_p,\, y_p,\, z_p\}$ are the ENU coordinates in meters.

However, this transformation is not enough for the intended application. The required position is of the CG, and the obtained position is from the receiver, that in the case of the FST06e is at a considerable distance. The general equation that corrects this offset is (6.2), where x_{rec} is the distance from the CG to the receiver.

$$\begin{bmatrix} p_x^{cg} \\ p_y^{cg} \end{bmatrix} = \begin{bmatrix} \cos\hat{\psi} & -\sin\hat{\psi} \\ \sin\hat{\psi} & \cos\hat{\psi} \end{bmatrix} \begin{bmatrix} x_{rec} \\ 0 \end{bmatrix} + \begin{bmatrix} x_p \\ y_p \end{bmatrix} \tag{6.2}$$

Since the position used is only in the X-Y plane, the vertical coordinate was drop. Also since the receivers are aligned with the X-axis, the y coordinate is assumed zero. The car GPS receiver is at $x_{rec} = 0.82m$ and the DGPS front receiver at $x_{rec} = 1.2m$

6.2.2 Accelerometer displacement

The accelerometer as stated in the model (2.34) is susceptible to the distance from the sensor location to the body CG, where the acceleration readings are required. For this correction is assumed that IMU location relative to CG is $\bar{d} = [0.46,\, 0,\, 0.2]\, m$.

6.3 Estimate results

Besides the corrections presented before, all the filters work and are used in the same way as in the simulation, with only small adjusts to the gains. In the following sections, the results of each filter will be presented and discussed. Whenever available, the data from the DGPS is presented side by side with estimated results. As stated before the DGPS system only works at 10Hz, and being a system that requires satellite data in urban environment is also susceptible to disturbances and errors. Nevertheless, the data retrieved are enough to work as ground truth for the estimates.

The data presented only goes to 300 seconds, since at that point the DGPS turned off. The filters continue but without ground truth for reference it was not worth it to presented.

It is noteworthy that the filter does not need the DGPS to work. It is only needed for comparison purposes.

6.3.1 Attitude Complementary Filter

In Fig.6.6 is depicted the yaw angle of the car, using the raw data from the magnetometer after the calibration proposed in Appendix B, the estimated yaw angle from the ACF and the yaw angle computed by the DGPS. The ACF gains used are presented in Tab.6.8 as well as the weights used for state and observations.

From the comparison of the magnetometer with the estimates is possible to see that the estimated is a much cleaner signal, and the effects of using the yaw rate in ACF are clearly visible. These influences can be spotted especially in turnings like between 200s and 202s, where the estimate is separated from the magnetometer and closer to the DGPS value. But in some other occasions the DGPS angle is much different from the estimate. However the DGPS values are not completely explainable, with

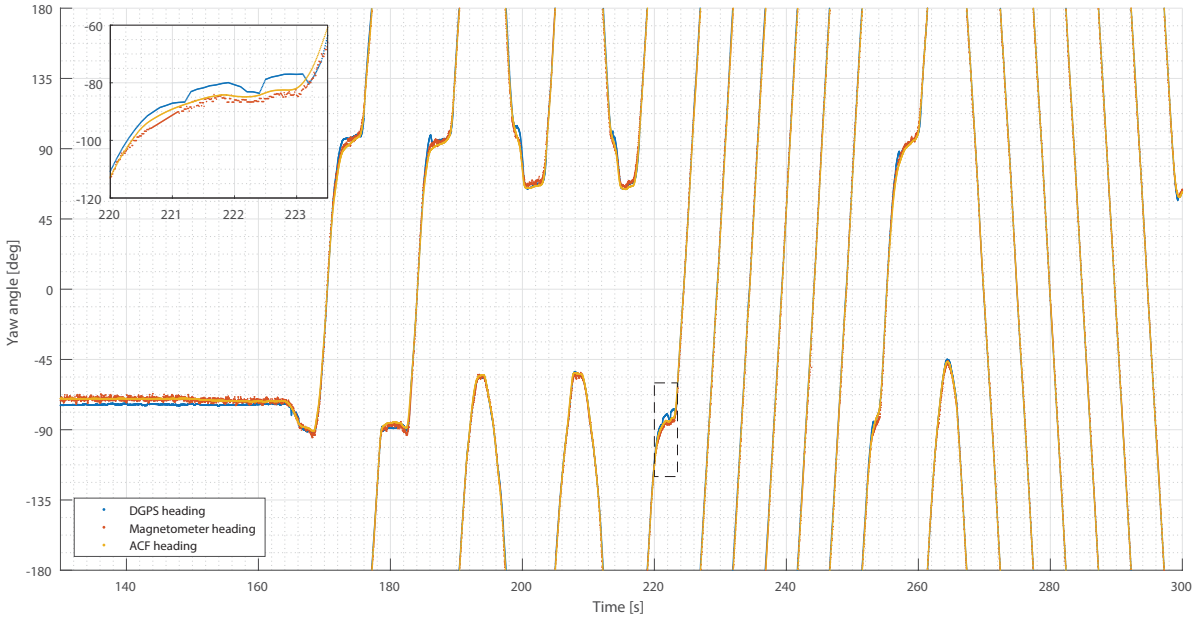

Figure 6.6: Yaw angle comparison between the DGPS, the magnetometer and the ACF estimate, also a detail of the yaw angle from 220 to 223.5 seconds.

Observation weight	State weight	Filter Gain
$R_{acf} = \begin{bmatrix} 10^3 \end{bmatrix}$	$Q_{acf} = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{bmatrix}$	$K_{acf} = \begin{bmatrix} 4.572 \times 10^{-3} \\ -9.977 \times 10^{-4} \end{bmatrix}$

Table 6.1: Parameters for the Attitude Complementary Filter used in the FST06e data.

some variations that are hard to justify like the detail between the 220s and the 223.5s where the magnetometer shows a smooth behaviour, the ACF estimate also shows a smooth behaviour, even thou it has a clear correction given by the yaw rate, and the DGPS has three major variations that do not appear in the previous two signals. However, in the beginning and at the end the three signals have closely the same values. These oscillations may be due to GPS acquisition difficulties at the time.

Even with these oscillations in Fig.6.7 is possible to see the angle error along the time to the DGPS signal from the raw angle and the estimated yaw. It is clear that even with some error due to the estimate or due to the DGPS, the yaw estimated by the ACF is more stable and smaller than the direct raw angle.

In Fig.6.8 is presented the yaw rate bias estimate for different initial conditions as well as the offset observed on the sensor output. As seen in the simulations, the roll, pitch and ground inclination influences the bias, which is evident when the car starts moving at the 162 seconds. Before that, the bias is stable around the offset.

In this filter since it has around 160 seconds to stabilize, the gain was computed iteratively along the simulation, which converges just after 15 seconds. The peak at the beginning is due to the initial error covariance matrix used. Even though these peaks are undesirable in a filter, since it had some time to stabilize, they are useful for helping in a faster convergence.

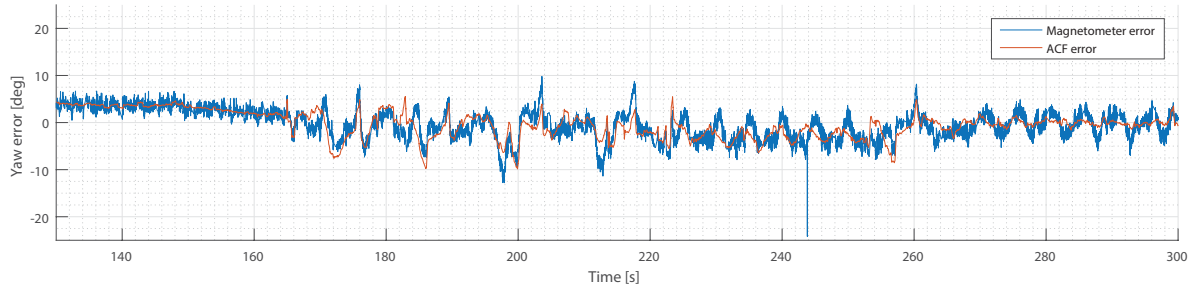

Figure 6.7: Yaw error comparison of the raw magnetometer yaw and the ACF estimate relative to the DGPS yaw angle.

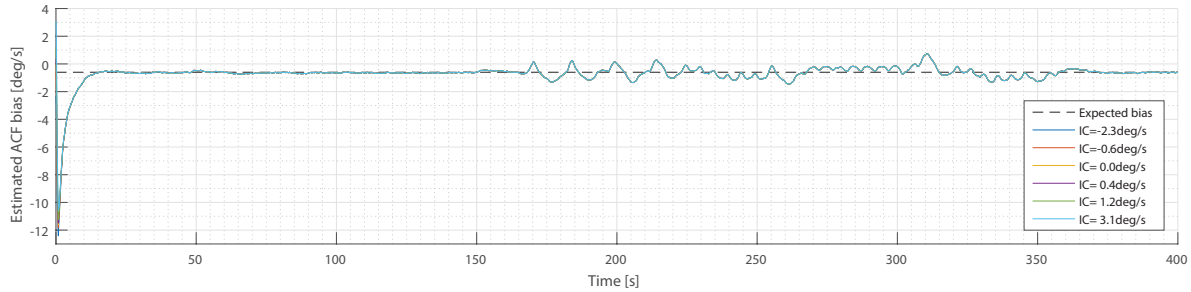

Figure 6.8: Bias estimate of ACF for the yaw rate, using different initial conditions (IC).

6.3.2 Position Complementary Filter

In Fig.6.9 is depicted the longitudinal velocity estimate by the PCF compared with the longitudinal velocity from the DGPS. Analogously in Fig.6.10 is depicted the same comparison but for the lateral velocity.

The weights used for the computation of the Kalman gain and the gain itself are presented in Tab.6.2. As the ACF, the Kalman gain for the PCF was computed iteratively along the time reaching the convergence for the K_{pcf} before the 40 seconds. As stated before, this method as some disadvantages like the transient part at the beginning of the estimates, but results in faster convergence of the values, especially the bias.

The gains used for the FST06e data presented where are different from the ones used in the simulation. In the simulation, was given a smaller weight (more confidence) to the state matrix, in this data the smallest weight was for the observation matrix. This was due to the accelerometer readings that were much noisier than the expected. This noise is small when the IMU is standing still, but when moving the noise increases. For example, in the constant velocity turn between 223s and 253s, where the lateral acceleration is expected to be constant, the mean value is $10m/s^2$ with a standard deviation of $1.8m/s^2$ much higher than the used in the simulation (Tab.4.2) of $0.7m/s^2$. These higher values force the filter to

Observation weight	State weight	Filter Gain
$R_{pcf} = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{bmatrix}$	$Q_{pcf} = \begin{bmatrix} \mathbf{I}_2 & 0 & 0 \\ 0 & 5\mathbf{I}_2 & 0 \\ 0 & 0 & 7\times10^{-3}\,\mathbf{I}_2 \end{bmatrix}$	$K_{pcf} = \begin{bmatrix} 0.999\,\mathbf{I}_2 & 0 & 0 \\ 0 & 2.25\mathbf{I}_2 & 0 \\ 0 & 0 & -0.082\mathbf{I}_2 \end{bmatrix}$

Table 6.2: Parameters for the Position Complementary Filter used in the FST06e data.

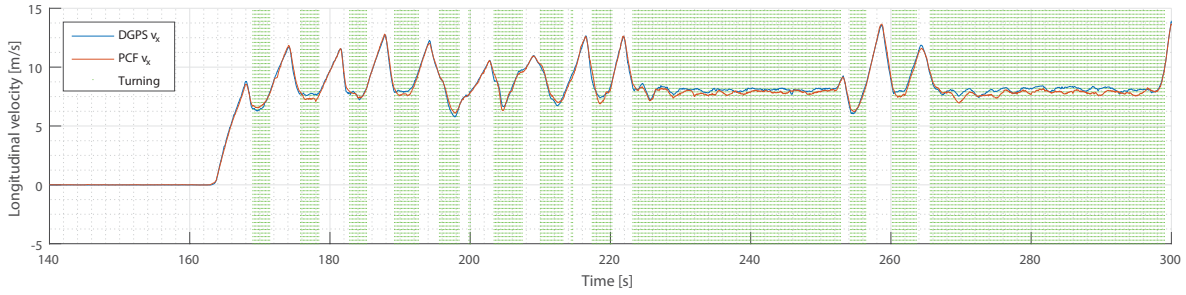

Figure 6.9: Longitudinal velocity results using the PCF relative to the DGPS longitudinal velocity. Also green zones represent periods of time when the yaw rate was $\geq$ 30deg/s.

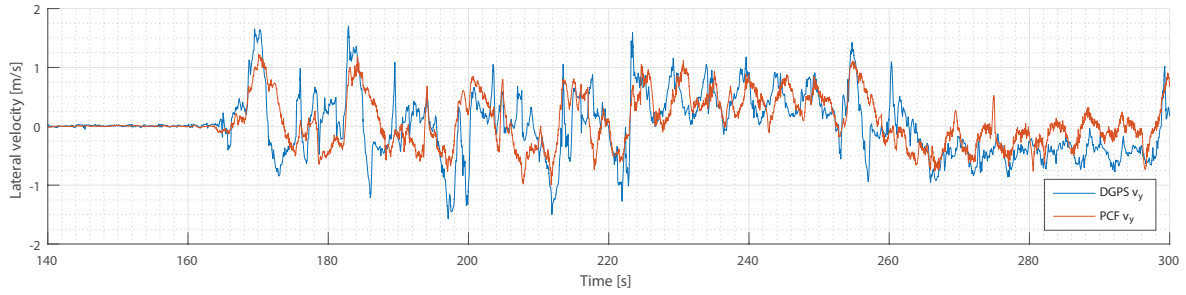

Figure 6.10: Lateral velocity results using the PCF relative to the DGPS lateral velocity.

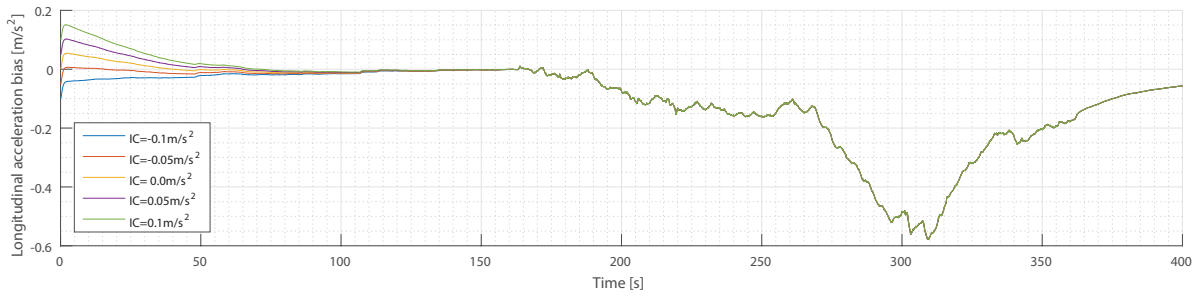

Figure 6.11: Bias estimate for the longitudinal acceleration using the PCF for different initial condition (IC).

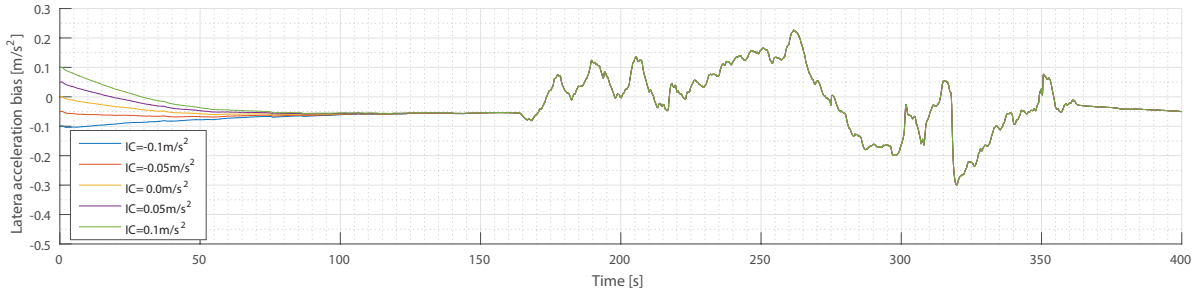

Figure 6.12: Bias estimate for the lateral acceleration using the PCF for different initial condition (IC).

have more confidence in the GPS position.

Analysing Fig.6.9 is possible to see that in general the estimate of the longitudinal velocity accompanies the longitudinal velocity from the DGPS, especially in straight lines. However, some discrepancies appear during the turns (green zones), generating some errors that can be neglected due to the range of velocities used.

In the lateral velocity, Fig.6.10, the estimated value suffers similar errors, but since the range used in the lateral velocity is smaller, the errors are more evident. The estimated lateral velocity shows a similar behaviour to the DGPS lateral velocity. Even in the constant velocity turns ([223,253] seconds and [266,299] seconds), is able to pick the subtle changes in the lateral velocities cause by the road inclination. However, at some other instants, both velocities have opposite behaviours, as for example in [206,210] seconds.

Like it was explained before, the bias of the accelerometer cannot be as easily predicted as the yaw rate, so in Fig.6.11 and Fig.6.12, the bias estimate of the longitudinal and lateral accelerations respectively, is presented for different initial conditions. In both cases is possible to see that the bias converges for a value when the car is stopped, but when the car is moving the bias tries to adapt. Note that the constant velocity turns mentioned before are in opposite directions.

6.3.3 Vehicle Linear Estimator and Vehicle Non-Linear Estimator

Next are presented the sideslip estimate results for the VLE (Fig.6.13) and VNLE (Fig.6.14), both of them compared with the sideslip from the DGPS. The results are divided in two graphs for an easier reading.

The weights used for the computation of the filter gains can be seen in Tab.6.3. Like before, the Kalman gain for the VLE was computed before relative to each longitudinal velocity and then used as a lookup table in the filter. The gains for the VNLE had to be computed online at each iteration. The major change to the weights was the observation matrix of the VNLE, where the gains were slightly increased. This was made to better adapt the filter to the measurements, but mainly due to the longitudinal force introduction, that has been assumed as a free noise signal, but like any measurement that is not true.

The three signals presented, VLE, VNLE and DGPS sideslip, show a perfect zero angle until the 164 seconds, this was due to post-processing where is assumed that below 3m/s of longitudinal velocity, no sideslip occur. This algorithm was also used in the simulation and has as its sole purpose to remove undesirable noise when the vehicle is standing still or slow enough to not generate significant sideslip. Note that the 3m/s isn't any special value, just a threshold that seem to not introduce any significant error, but that could do the job of cleaning the noise.

As already seen before, the VLE shows up as a smoother signal than the VNLE, however it is not as close to the DGPS signal.

Analysing first the VLE in Fig.6.3 shows that almost every time that a clear spike of sideslip appears in the DGPS, the VLE also picks it up. However, the estimated angle is a little more conservative than the one from the DGPS. Besides that, in the constant velocity turns, even with the lateral velocity observation oscillating due to the road inclination, the VLE sideslip is almost constant.

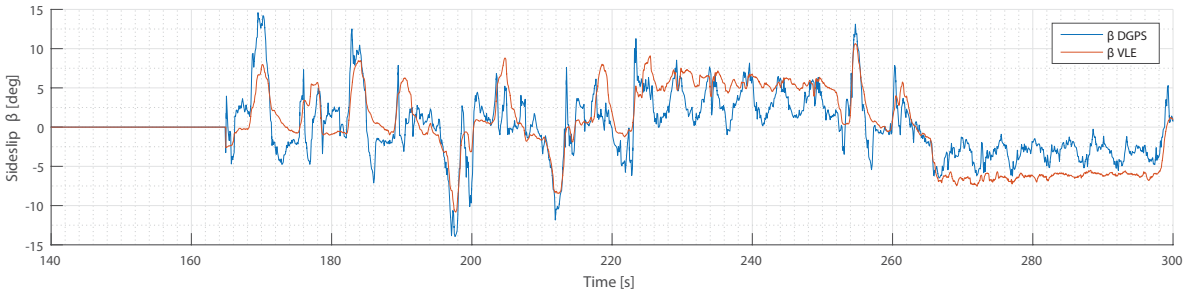

Figure 6.13: Comparison between the sideslip β using the VLE and the β computed from the DGPS.

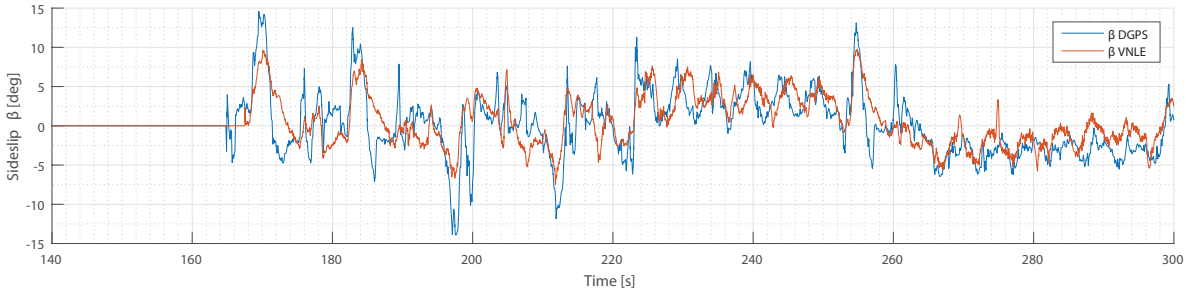

Figure 6.14: Comparison between the sideslip β using the VNLE and the β computed from the DGPS.

	VLE	VNLE
Observation weight	$R_{VLE} = \begin{bmatrix} 3 \times 10^{-2} & 0 \\ 0 & 5 \times 10^{-2} \end{bmatrix}$	$R_{VNLE} = \begin{bmatrix} 2 \times 10^{-3} & 0 & 0 \\ 0 & 2 \times 10^{-3} & 0 \\ 0 & 0 & 10^{-2} \end{bmatrix}$
State weight	$Q_{VLE} = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-4} \end{bmatrix}$	$Q_{VNLE} = \begin{bmatrix} 10^{-1} & 0 & 0 \\ 0 & 10^{-3} & 0 \\ 0 & 0 & 9 \times 10^{-1} \end{bmatrix}$
Covariance error	$P_{VLE} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$	$P_{VNLE} = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}$

Table 6.3: Parameters for the computation of Kalman gains for both the VLE and the VNLE used with the FST06e.

The VNLE, unlike the VLE, can catch smaller disturbances at the cost of being noisier. Where the VNLE stands out is the constant velocity turns, where it can pick the oscillations induced by the road inclination something that the VLE cannot do. Another situation that attracts some attention is around the second 218, where the VNLE and the DGPS have opposite reactions. This situation also emerged in the lateral velocity estimate in Fig.6.10, and arise a doubt of whose fault is it. It can be that the VNLE is using two much of the PCF estimate, and this one had a bad estimation for that point for some reason, or it can be an error in the DGPS like those seen in the ACF.

Other situation that arises, is in the constant velocity turns, where the VNLE and the DGPS seem unsynchronized. This may be related to the time log of some system that for some reason got delayed. In the overall is safe to conclude that the VNLE had better results compared with the VLE.

Without an absolute, high frequency sensor for the sideslip like the optical flow, is not possible to clearly say if the estimator architecture used in the car could complete the job with small errors or not. However, with the estimated value close to the DGPS sideslip as seen with the VNLE, is possible to say that it is a good alternative system, and with a more deeper validation, and more accurate sensors could reveal as the best overall alternative. Additionally, in Tab.6.4 is possible to see the statistical values of error for both vehicle estimators.

	Max Error	mean	σ	σ^2
VLE	9.3041	-0.0763	2.7314	7.4606
VNLE	11.1352	-0.3845	2.5833	6.6736

Table 6.4: Comparison of statistical data for the computation of the sideslip angle for the result with the VLE and VNLE.

Chapter 7

Conclusions

7.1 Achievements

This works proposes an estimator architecture using a fusion of a GPS/INS to retrieve a sideslip estimation of a Formula Student prototype. The main goal is to have an on-board system capable of providing real-time data at high enough frequencies to be used in a more complex control algorithm and, in this way, be able to replace expensive dedicated sensors.

First a complex vehicle model was presented, that can recreate the non-linearities and interactions of the different modules. This model is used to simulate the data acquired from sensors on-board the vehicle, and provide the true result of sideslip.

An architecture was presented using three filters. Two kinematic, one for the attitude that combines the readings of a digital compass with the yaw rate, giving a heading angle of the vehicle. A second that uses the position from a GPS with the accelerometers on-board to provide the velocity components in the vehicle frame.

For the third filter it is used the planar vehicle model to introduce the dynamics of the car and provide the sideslip angle. For this one, two estimators are compared one with linear dynamics, and other with non-linear dynamics. The linear estimator showed good results for a more controlled and predicted actuation, as the non-linear showed good results for an overall actuation with the price of being computationally heavier.

The estimator architecture was then tested with real data. First, inside the laboratory where the filters were implemented in a micro-controller and running data at 100Hz, proving that the system was capable of working in the necessary conditions to supply a control algorithm. Then, the estimator architecture was tested off-line, but with real data acquired from a Formula Student prototype where it was compared side-by-side with the data from a DGPS, showing very promising results.

From this work resulted two technical conference papers. One already presented [46], and a second [47] already accepted and to be presented in the end of November 2017.

7.2 Future Work

Despite the developed work achieved almost every main objective proposed, some work should still be done to improve this system.

The missing objective in this work, is the implementation in real-time in a Formula Student prototype. The work addressed the implementation of the filters in a micro-controller, and tested the architecture with real data, but was still missing the opportunity to be implemented in the vehicle.

The results obtained can still be improved. The IMU used is a low-cost one, and the data retrieved from it is not ideal for this type of work. The replacing of the IMU for one more precise and accurate is a must in a system like this.

Another route to be explored, is the use of redundant sensors, namely, the use of two IMUs in different locations of the vehicle.

In this work, was always possible to use all the sensors at the same frequency. The use of sensors with different rates should also be explored.

Even after all the work done with the calibration of the magnetometer, this sensor was always the weaker point of the system. A more reliable source for the heading should be explored since the remaining filters depend widely on this information.

Bibliography

[1] W. Bertelsen. *Popular Science*, chapter You'll cross the U.S. in two hours in the air car that steers and parks itself, pages 68, 69, 178. August 1961.

[2] L. B. Almeida, J. Azevedo, C. Cardeira, P. Costa, P. Fonseca, P. Lima, A. F. Ribeiro, and V. Santos. Mobile robot competitions: fostering advances in research, development and education in robotics. 2000.

[3] SAE International. Automated driving: Levels of driving automation are defined in new sae international standard J3016., 2014. URL `https://www.sae.org/misc/pdfs/automated_driving.pdf`. Accessed: 2017-09-30.

[4] L. De Novellis, A. Sorniotti, P. Gruber, and A. Pennycott. Comparison of feedback control techniques for torque-vectoring control of fully electric vehicles. *IEEE Transactions on Vehicular Technology*, 63(8):3612–3623, 2014.

[5] M. Doumiati, A. C. Victorino, A. Charara, and D. Lechner. Onboard real-time estimation of vehicle lateral tire–road forces and sideslip angle. *IEEE/ASME Transactions on Mechatronics*, 16(4):601–614, 2011.

[6] D. Bevly, J. Ryu, and J. Gerdes. Integrating ins sensors with gps measurements for continuous estimation of vehicle sideslip, roll, and tire cornering stiffness. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):483–493, December 2006.

[7] H. F. Grip, L. Imsland, T. A. Johansen, J. C. Kalkkuhl, and A. Suissa. Vehicle sideslip estimation. *IEEE control systems*, 29(5), 2009.

[8] J. Ryu, E. J. Rossetter, and J. C. Gerdes. Vehicle sideslip and roll parameter estimation using gps. In *Proceedings of the AVEC International Symposium on Advanced Vehicle Control*, 2002.

[9] M. Doumiati, A. Victorino, A. Charara, and D. Lechner. A method to estimate the lateral tire force and the sideslip angle of a vehicle: Experimental validation. In *American Control Conference (ACC), 2010*, pages 6936–6942. IEEE, 2010.

[10] M. Gadola, D. Chindamo, M. Romano, and F. Padula. Development and validation of a kalman filter-based model for vehicle slip angle estimation. *Vehicle System Dynamics*, 52(1):68–84, 2014.

[11] K. Nam, S. Oh, H. Fujimoto, and Y. Hori. Estimation of sideslip and roll angles of electric vehicles using lateral tire force sensors through rls and kalman filter approaches. *IEEE Transactions on Industrial Electronics*, 60(3):988–1000, March 2013.

[12] C. Geng, L. Mostefai, M. Denaï, and Y. Hori. Direct yaw-moment control of an in-wheel-motored electric vehicle based on body slip angle fuzzy observer. *IEEE Transactions on Industrial Electronics*, 56(5):1411–1419, 2009.

[13] M. Abdulrahim. On the dynamics of automobile drifting. Technical report, SAE Technical Paper, 2006.

[14] P. Zhang, J. Gu, E. E. Milios, and P. Huynh. Navigation with imu/gps/digital compass with unscented kalman filter. In *Mechatronics and Automation, 2005 IEEE International Conference*, volume 3, pages 1497–1502. IEEE, 2005.

[15] J. Sasiadek and P. Hartana. Sensor data fusion using kalman filter. In *Information Fusion, 2000. FUSION 2000. Proceedings of the Third International Conference on*, volume 2, pages WED5–19. IEEE, 2000.

[16] A. Nemra and N. Aouf. Robust ins/gps sensor fusion for uav localization using sdre nonlinear filtering. *IEEE Sensors Journal*, 10(4):789–798, 2010.

[17] S. Sukkarieh, E. M. Nebot, and H. F. Durrant-Whyte. A high integrity imu/gps navigation loop for autonomous land vehicle applications. *IEEE Transactions on Robotics and Automation*, 15(3):572–578, 1999.

[18] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel. A complementary filter for attitude estimation of a fixed-wing uav. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 340–345. IEEE, 2008.

[19] J. F. Vasconcelos, B. Cardeira, P. Oliveira, and P. Batista. Discrete-time complementary filters for attitude and position estimation: Design, analysis and experimental validation. *IEEE Transactions on Control Systems Technology*, 19(1):181–198, January 2011.

[20] N. Martins. Integration of rc vehicles in a robotic arena. Master's thesis, Instituto Superior Técnico, November 2016.

[21] C. C. de Wit and P. Tsiotras. Dynamic tire friction models for vehicle traction control. In *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No.99CH36304)*, volume 4, pages 3746–3751 vol.4, 1999.

[22] H. Pacejka. *Tire and Vehicle Dynamics*. Butterworth-Heinemann, 2^{nd} edition, 2006. ISBN:980-0750669184.

[23] W. Hirschberg, G. Rill, and H. Weinfurter. Tire model tmeasy. *Vehicle System Dynamics*, 45(sup1): 101–119, 2007.

[24] U. Kiencke and L. Nielsen. *Automotive Control Systems for Engine, Driveline, and Vehicle*. Springer, 2^{nd} edition, 2005. ISBN:978-3540231394.

[25] H. B. Pacejka and E. Bakker. The magic formula tyre model. *Vehicle System Dynamics*, 21(sup001): 1–18, 1992.

[26] R. N. Jazar. *Vehicle Dynamics: Theory and Applications*. Springer, 1^{st} edition, 2008. ISBN:978-0387742434.

[27] E. Kasprzak and D. Gentz. The formula sae tire test consortium-tire testing and data handling. *SAE Paper 2006-01-3606, Society of Automotive Engineers*, 2006.

[28] W. Milliken and D. Milliken. *Race Car Vehicle Dynamics*. Society of Automotive Engineers Warrendale, 1995. ISBN:156-0915299.

[29] M. Blundell and D. Harty. *The Multibody Systems Approach to Vehicle Dynamics*. Butterworth-Heinemann, 2004. ISBN:0-7506-5112-1.

[30] B. Cardeira. Arquitecturas para navegação inercial/gps com aplicação a veículos autónomos. Master's thesis, Instituto Superior Técnico, February 2009.

[31] J. Vasconcelos, G. Elkaim, C. Silvestre, P. Oliveira, and B. Cardeira. Geometric approach to strapdown magnetometer calibration in sensor frame. *IEEE Transactions on Aerospace and Electronic Systems*, 47(2):1293–1306, April 2011.

[32] R. M. Rogers. *Applied Mathematics in Integrated Navigation Systems*. AIAA Education Series, 3^{rd} edition, 2000.

[33] J. C. Dixon. *Suspension Geometry and Computation*. Wiley, 1^{st} edition, 2009. ISBN:978-0470510216.

[34] R. E. Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[35] B. Friedland. *Control system design: an introduction to state-space methods*. Courier Corporation, 2012.

[36] R. G. Brown and P. Hwang. *Introduction to random signals and applied Kalman filtering: with MATLAB exercises*. Wiley, 4^{th} edition, 2012. ISBN:978-0470609699.

[37] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, 1979. 0-13-638122-7.

[38] F. Lewis, D. Vrabie, and V. Syrmos. *Optimal Control*. Wiley, 3^{rd} edition, 2012. ISBN 978-0470633496.

[39] M. I. Ribeiro. Kalman and extended kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics*, 43, 2004.

[40] A. Gelb. *Applied optimal estimation*. MIT press, 1974.

[41] W. J. Rugh. *Linear System Theory*. Prentice-Hall, 2^{nd} edition, 1996.

[42] R. DeCarlo. *Linear Systems: A State Variable Approach with Numerical Implementation*, page 215. Prentice-Hall, 1989.

[43] K. Reif, S. Günther, E. Yaz, and R. Unbehauen. Stochastic stability of the discrete-time extended kalman filter. *IEEE Transactions on Automatic Control*, 44(4):714–728, April 1999.

[44] Wiring Pi: GPIO interface library for the raspberry pi. URL `http://wiringpi.com`. Last accessed: 2017-09-30.

[45] *Datum Transformations of GPS Positions: Application Note*. µ-blox ag, July 1999. URL `http://www.nalresearch.com/files/Standard%20Modems/A3LA-XG/A3LA-XG%20SW%20Version%201.0.0/GPS%20Technical%20Documents/GPS.G1-X-00006%20(Datum%20Transformations).pdf`.

[46] A. Antunes, C. Cardeira, and P. Oliveira. Sideslip estimation of formula student prototype through GPS/INS fusion. In *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 184–191. IEEE, 2017.

[47] A. Antunes, C. Cardeira, and P. Oliveira. Application of sideslip estimation architecture to a formula student prototype. In *ROBOT'2017 Third Iberian Robotics Conference*, Seville, Spain, 22-24 November 2017.

[48] A. Fitzgibbon, M. Pilu, and R. B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):476–480, 1999.

Appendix A

Vertical Model Expansion

The model presented in section 2.2 for the vertical dynamics of the car is defined has a classic state-space system $\dot{\mathbf{x}} = A.\mathbf{x} + B.\mathbf{u}$ where $\mathbf{x}$ is the vector containing the state variables (A.1), and $\mathbf{u}$ is the input vector (A.2).

$$\mathbf{x} = \begin{bmatrix} Z & \dot{Z} & \theta & \dot{\theta} & \phi & \dot{\phi} & h_1 & \dot{h}_1 & h_2 & \dot{h}_2 & h_3 & \dot{h}_3 & h_4 & \dot{h}_4 \end{bmatrix}^T \tag{A.1}$$

$$\mathbf{u} = \begin{bmatrix} F_z & M_\theta & M_\phi & G_1 & G_2 & G_3 & G_4 \end{bmatrix} \tag{A.2}$$

The expanded model equations from section 2.2:

- The expansion of equation (2.7) for the Z-axis:

$$\begin{aligned} m_{ch}\ddot{Z} =& F_z - [K_{s1} + K_{s2} + K_{s3} + K_{s4}]\, Z - [C_{s1} + C_{s2} + C_{s3} + C_{s4}]\, \dot{Z} \\ &+ [(K_{s2} + K_{s4})d - (K_{s1} + K_{s3})c]\, \phi + [(C_{s2} + C_{s4})d - (C_{s1} + C_{s3})c]\, \dot{\phi} \\ &+ [(K_{s1} + K_{s2})a - (K_{s3} + K_{s4})b]\, \theta + [(C_{s1} + C_{s2})a - (C_{s3} + C_{s4})b]\, \dot{\theta} \\ &+ K_{s1}h_1 + C_{s1}\dot{h}_1 + K_{s2}h_2 + C_{s2}\dot{h}_2 + K_{s3}h_3 + C_{s3}\dot{h}_3 + K_{s4}h_4 + C_{s4}\dot{h}_4 \end{aligned} \tag{A.3}$$

- The expansion of equation (2.8) for the roll:

$$\begin{aligned} I_\phi\ddot{\phi} =& M_\theta + [(K_{s2} + K_{s4})d - (K_{s1} + K_{s3})c]\, Z + [(C_{s2} + C_{s4})d - (C_{s1} + C_{s3})c]\, \dot{Z} \\ &- \left[(K_{s1} + K_{s3})c^2 + (K_{s2} + K_{s4})d^2\right] \phi - \left[(C_{s1} + C_{s3})c^2 + (C_{s2} + C_{s4})d^2\right] \dot{\phi} \\ &+ [(K_{s1}a - K_{s3}b)c + (K_{s4}b - K_{s2}a)d]\, \theta + [(C_{s1}a - C_{s3}b)c + (C_{s4}b - C_{s2}a)d]\, \dot{\theta} \\ &+ K_{s1}ch_1 + C_{s1}c\dot{h}_1 - K_{s2}dh_2 - C_{s2}d\dot{h}_2 + K_{s3}ch_3 + K_{s3}c\dot{h}_3 - K_{s4}dh_4 - K_{s4}d\dot{h}_4 \end{aligned} \tag{A.4}$$

- The expansion of equation (2.9) for the pitch:

$$\begin{aligned} I_\theta\ddot{\theta} =& M_\theta + [(K_{s1} + K_{s2})a - (K_{s3} + K_{s4})b]\, Z + [(C_{s1} + C_{s2})a - (C_{s3} + C_{s4})b]\, \dot{Z} \\ &+ [(K_{s1}c - K_{s2}d)a + (K_{s4}d - K_{s3}c)b]\, \phi + [(C_{s1}c - C_{s2}d)a + (C_{s4}d - C_{s3}c)b]\, \dot{\phi} \\ &- \left[(K_{s1} + K_{s2})a^2 + (K_{s3} + K_{s4})b^2\right] \theta - \left[(C_{s1} + C_{s2})a^2 + (C_{s3} + C_{s4})b^2\right] \dot{\theta} \\ &- K_{s1}ah_1 - C_{s1}a\dot{h}_1 - K_{s2}ah_2 - C_{s2}a\dot{h}_2 + K_{s3}bh_3 + C_{s3}b\dot{h}_3 + K_{s4}bh_4 + C_{s4}b\dot{h}_4 \end{aligned} \tag{A.5}$$

- The expansion of equation (2.10) for the unsprung mass 1:

$$\begin{aligned} m_{u1}\ddot{h}_1 =& K_{s1}Z + C_{s1}\dot{Z} + K_{s1}c\phi + C_{s1}c\dot{\phi} - K_{s1}a\theta - C_{s1}a\dot{\theta} \\ & - \left[K_{s1} + K_{p1} + K_{ar1}\right]h_1 - \left[C_{s1} + C_{p1}\right]\dot{h}_1 + K_{ar1}h_2 + K_{p1}G_1 \end{aligned} \tag{A.6}$$

- The expansion of equation (2.11) for the unsprung mass 2:

$$\begin{aligned} m_{u2}\ddot{h}_2 =& K_{s2}Z + C_{s2}\dot{Z} - K_{s2}d\phi - C_{s2}d\dot{\phi} - K_{s2}a\theta - C_{s2}a\dot{\theta} \\ & - \left[K_{s2} + K_{p2} + K_{ar1}\right]h_2 - \left[C_{s2} + C_{p2}\right]\dot{h}_2 + K_{ar1}h_2 + K_{p2}G_2 \end{aligned} \tag{A.7}$$

- The expansion of equation (2.12) for the unsprung mass 3:

$$\begin{aligned} m_{u3}\ddot{h}_3 =& K_{s3}Z + C_{s3}\dot{Z} + K_{s3}c\phi + C_{s3}c\dot{\phi} + K_{s3}b\theta + C_{s3}b\dot{\theta} \\ & - \left[K_{s3} + K_{p3} + K_{ar2}\right]h_3 - \left[C_{s3} + C_{p3}\right]\dot{h}_3 + K_{ar2}h_3 + K_{p3}G_3 \end{aligned} \tag{A.8}$$

- The expansion of equation (2.13) for the unsprung mass 4:

$$\begin{aligned} m_{u4}\ddot{h}_4 =& K_{s4}Z + C_{s4}\dot{Z} - K_{s4}d\phi - C_{s4}cd\dot{\phi} + K_{s4}b\theta + C_{s4}b\dot{\theta} \\ & - \left[K_{s4} + K_{p4} + K_{ar2}\right]h_4 - \left[C_{s4} + C_{p4}\right]\dot{h}_4 + K_{ar2}h_4 + K_{p4}G_4 \end{aligned} \tag{A.9}$$

Appendix B

Magnetometer Calibration

The magnetometer is used to acquire the yaw angle of the vehicle, but its readings are full of errors and perturbation that must be corrected before the data can be used. In this chapter, a small introduction will be made to the errors associated with this sensor as well as a proposed model by [30]. Later on, keeping in mind the sensor model, a calibration procedure will be presented using real data, where the results can be integrated in a real-time acquisition system, to produce a more accurate yaw angle measure.

B.1 Magnetometer model

The magnetometer suffers from several sources of error being them:

- Magnetic declination - Since the sensor is to be used in a vehicle, in a track, the magnetic declination variation is minimal, and can be considered just a constant deviation in the yaw reading.
- Hard Iron - This is created by ferromagnetic materials, and in general by any material in the vehicle or in the environment that generates a constant magnetic field additive to the earth's magnetic field. Thus, usually represented by an offset associated to the magnetic readings.
- Soft Iron - This is a distortion to the magnetic field created by materials that do not necessarily generate magnetic fields. These materials distort the earth magnetic field and the fields generated by other materials.
- Electric influences - Nearby electric elements like electric motors and just the current flowing in a wire or circuit, generates magnetic fields that add to the earth magnetic field. Unlike the hard iron influence that is constant, current flowing in a motor is much motor unpredictable, since the magnetic field will change with velocity and torque. The same goes for current in a wire or circuit.
- Manufacturing and application - Errors or imprecisions in the manufacturing of the sensor can generate misalignments between the axes, and different scale factors for each axis. Also, when soldering and applying a sensor, misalignments between the sensor and vehicle frames also generate errors.

- Noise - All sensors have noise, and the magnetometer is no exception.

Is almost impossible to correct all the errors perfectly, but is possible to reduce their influence. In order to calibrate this sensor, first is necessary to comprehend how the above distortions influence the readings. A model for the magnetometer is proposed by [30] and [31] described by equation (B.1). Where $\mathbf{m}$ represents the ideal magnetic vector, and $\mathbf{m}_r$ the sensor reading.

$$\mathbf{m}_r = R_\eta^T C_{SF} R_\eta \left(C_{SI}\mathbf{m} + \delta_{HI} \right) \tag{B.1}$$

As stated before, the hard iron distortion is an additive magnetic field represented by δ_{HI}, and the soft iron distortion are represented by the C_{SI} matrix. The hard and soft iron distortions equations are presented in more detail in (B.2). Where the terms c_{SI_1}, c_{SI_2}, c_{SI_3}, c_{SI_4}, c_{SI_5}, c_{SI_6}, c_{SI_7}, c_{SI_8}, c_{SI_9} are responsible to generate the soft iron effects.

$$C_{SI}\mathbf{m} + \delta_{HI} = \begin{bmatrix} (1 + c_{SI_1}) & c_{SI_2} & c_{SI_3} \\ c_{SI_4} & (1 + c_{SI_5}) & c_{SI_6} \\ c_{SI_7} & c_{SI_8} & (1 + c_{SI_9}) \end{bmatrix} \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} + \begin{bmatrix} \delta_{HI_x} \\ \delta_{HI_y} \\ \delta_{HI_x} \end{bmatrix} \tag{B.2}$$

The rotation matrix R_η, makes the rotation from the sensor frame to the body frame, and the matrix C_{SF} contains the scaling factors for each axis.

B.2 Calibration algorithm

The proposed algorithm for the magnetometer calibration, uses as reference the magnetometer model (B.1) and the calibration proposed by [30]. Also, some additional conditions were applied for the specific type of use needed. The following algorithm is exemplified using real data acquired using

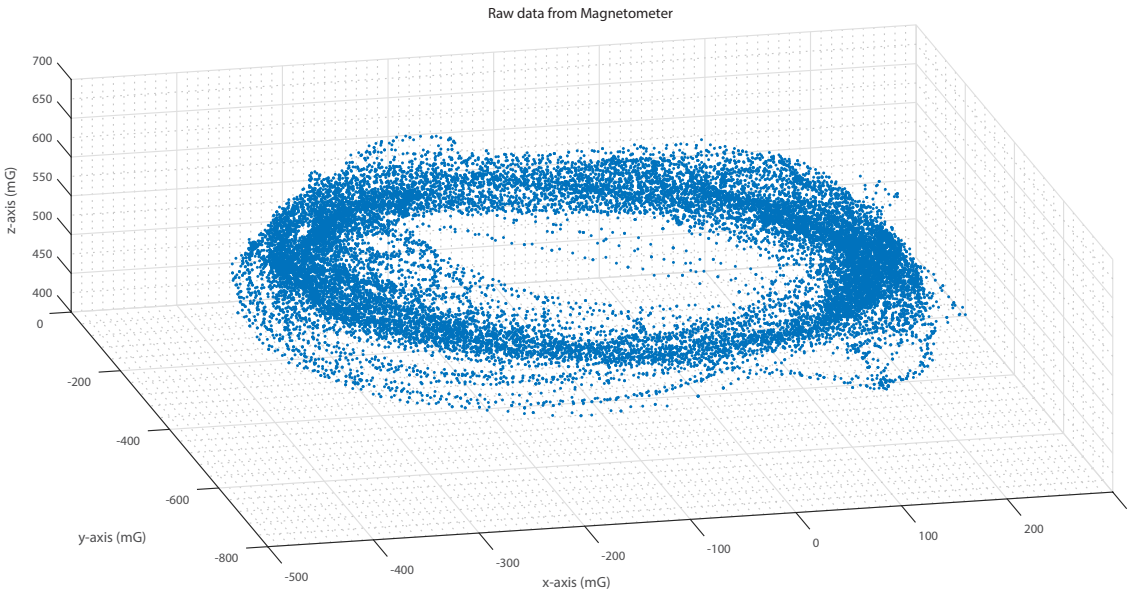

Figure B.1: Raw data from 3-axis magnetometer, acquired inside a laboratory

the RC car (Fig.5.1) inside a laboratory.

From Fig.B.1, is possible to see that the data is not usable as it is, being the major problem the centre of the axis. Computing the yaw like this will not return any usable data.

The first step is a fit to a plane in order to remove the IMU orientation, and application errors of the sensor. This can be done since the vehicle only moves in a plane, and the data used must be acquired in a flat surface. The raw data is fitted to a plane equation (B.3) as seen in (Fig.B.2) using a Least Squares algorithm with no weights.

$$P_1 x + P_2 y + P_3 = z \tag{B.3}$$

With the parameters P_1, P_2 and P_3, of the equation is then possible, using the normal vector to the plane, to compute the rotation matrices around the x and y axis respectively R_ϕ and R_θ. Being the rotation matrices for this given by (B.4) where the angles are computed from (B.5).

$$R_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}, \quad R_\theta = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{B.4}$$

$$\phi = \text{tg}^{-1}\left(\frac{P_2}{-1}\right), \quad \theta = \text{tg}^{-1}\left(\frac{P_1}{P_2^2 + (-1)^2}\right) \tag{B.5}$$

In Fig.B.3 is possible to see the influence of these rotation. For this case, the differences are small since the IMU frame is pretty close to the vehicle frame.

The z-axis of the sensor is only needed for the calibration above. Since the yaw only depends of the x and y axis, the z will be neglected from now on. In Fig.B.4 is possible to see the data corrected above in the x-y plane, and once again is clear that computing the yaw from this will have no utility.

The second step is a fit to a 2D ellipse. It's chosen an ellipse due to the transformation suffered from the soft iron distortions. The data is fitted to the conic equation of the ellipse (B.6), where E_1, E_2, E_3,

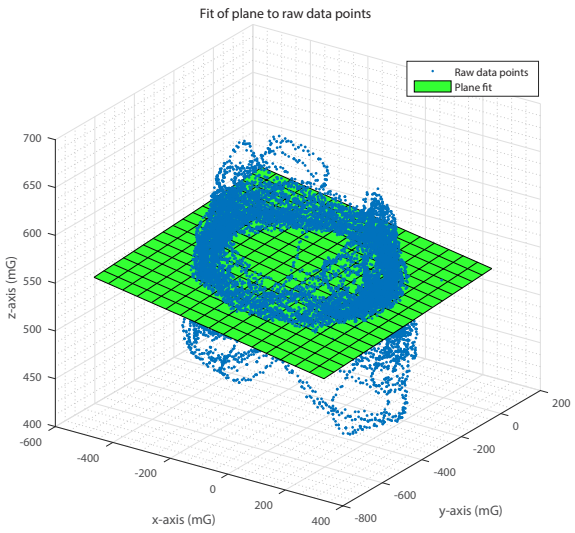

Figure B.2: Plane fitted to magnetometer raw data using least squares.

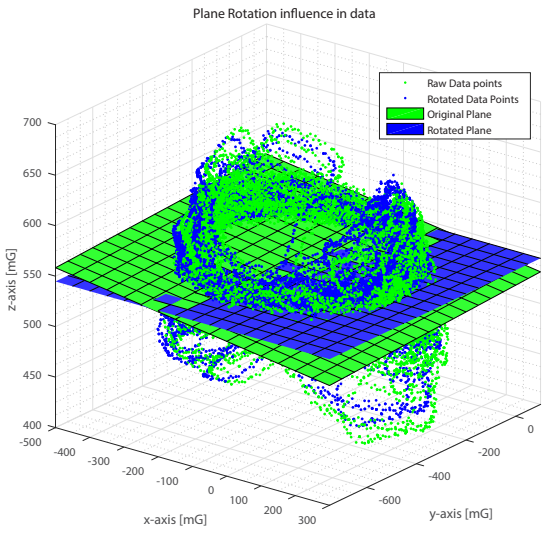

Figure B.3: Rotation of the fitted plane and data to an horizontal plane.

E_4, E_5, E_6 are coefficients to be determined using a Least Squares fitting [48]. For the sake of simplicity, is assumed $F = -1$. The fitted equation can be seen in Fig.B.4.

$$E_1x^2 + E_2xy + E_3y^2 + E_4x + E_5y + E_6 = 0 \tag{B.6}$$

As can easily be interpreted, the E_2 term is the responsible for the rotation of the ellipse. The non-tilted ellipse will have $E_2 = 0$, and to find it using equation (B.6), x is replaced by $x\cos\eta - y\sin\eta$ and y by $x\sin\eta - y\cos\eta$, where η is the ellipse angle of rotation. Rearranging the equation to be similar to the original results in (B.7).

$$\begin{aligned} & x^2\left(E_1\cos^2\eta - E_2\cos\eta\sin\eta + E_3\sin^2\eta\right) \\ & \quad + xy\left(2E_1\cos\eta\sin\eta + E_2\left(\cos^2\eta - \sin^2\eta\right) - 2E_3\cos\eta\sin\eta\right) \\ & \quad + y^2\left(E_1\sin^2\eta + E_2\cos\eta\sin\eta + E_3\cos^2\eta\right) \\ & \quad + x\left(E_3\cos\eta - E_5\sin\eta\right) + y\left(E_4\sin\eta + E\cos\eta\right) + E_6 = 0 \end{aligned} \tag{B.7}$$

Using the condition of no tilt $E_2 = 0$, is then easy to find the angle η from (B.8). With the angle found, the new coefficients for the non-tilted ellipse $\{n\}$ are given by (B.9)

$$2E_1\cos\eta\sin\eta + E_2\left(\cos^2\eta - \sin^2\eta\right) - 2E_3\cos\eta\sin\eta = 0 \quad \Rightarrow \quad \eta = \frac{1}{2}\,\mathrm{tg}^{-1}\left(\frac{E_2}{E_3 - E_1}\right) \tag{B.8}$$

$$\begin{aligned} {}^nE_1 &= E_1\cos^2\eta - E_2\cos\eta\sin\eta + E_3\sin^2\eta \quad , \qquad {}^nE_4 = \quad E_3\cos\eta - E_5\sin\eta \\ {}^nE_3 &= E_1\sin^2\eta + E_2\cos\eta\sin\eta + E_3\cos^2\eta \quad , \qquad {}^nE_5 = \quad E_4\sin\eta + E_5\cos\eta \end{aligned} \tag{B.9}$$

Using these new parameters, the conic equation of the ellipse (B.7) is transformed in the general equation of the ellipse (B.10) using the complete squares method, and the ellipse parameters retrieved as (B.11). Where x_0 and y_0 are the centre coordinates of the ellipse, r_a is the radius of axis a and r_b the radius of the axis b.

$$\frac{(x - x_0)^2}{r_a^2} + \frac{(y - y_0)^2}{r_b^2} = 1 \tag{B.10}$$

$$\begin{gathered} x_0 = \frac{-{}^nE_4}{2{}^nE_1} \quad , \qquad y_0 = \frac{-{}^nE_5}{2{}^nE_3} \quad , \qquad r_a = \sqrt{\left|\frac{{}^nE_6^{"}}{{}^nE_1}\right|} \quad , \qquad r_b = \sqrt{\left|\frac{{}^nE_6^{"}}{{}^nE_3}\right|} \\ where: \quad {}^nE_6^{"} = -E_6 + \frac{{}^nE_4^2}{4{}^nE_1} + \frac{{}^nE_5^2}{4{}^nE_3} \end{gathered} \tag{B.11}$$

With these parameters, the non-tilted ellipse centre is used to remove the offset, and the radius used to compute the scaling factor, σ_{sf}, that transforms the ellipse in a circle.

$$\sigma_{sf} = \frac{r_b}{r_a} \tag{B.12}$$

The result of the calibration compared with the initial raw data of the magnetometer can be seen in Fig.B.5. In this case the effect of the soft iron influences is not evident and as such, the data is close to a circle. This was also confirmed by the scale factor which was close to one, $\sigma_{sf} = 1.003$. Using

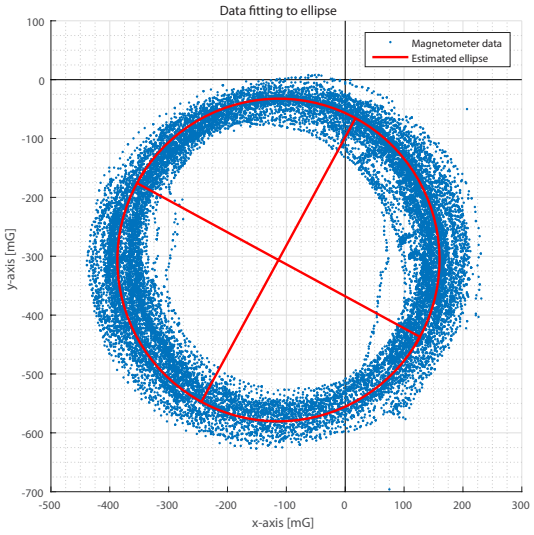

Figure B.4: Fitted ellipse to magnetometer data. The red lines represent the ellipse axis.

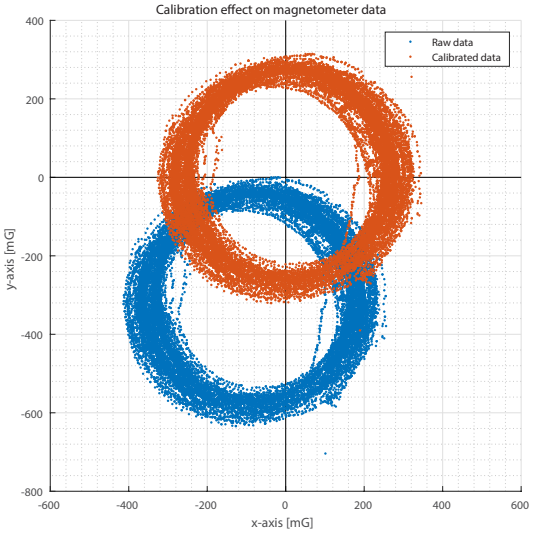

Figure B.5: Comparison between raw data, and calibrated.

all the constants found in this calibration/identification process, is possible to implement it in a real time acquisition using equation (B.13) which is similar to the previous model (B.1).

$$\mathbf{m}_C = R_\eta^T \left(\begin{bmatrix} \sigma_{sf} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \left[R_\eta \left(R_\theta R_\phi \mathbf{m}_r - R_\eta^T \begin{bmatrix} x_0 \\ y_0 \\ 0 \end{bmatrix} \right) \right] \right) \quad , \quad R_\eta = \begin{bmatrix} \cos\eta & -\sin\eta & 0 \\ \sin\eta & \cos\eta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{B.13}$$

Where $\mathbf{m}_r$ is the raw data readings from the magnetometer, R_x and R_y are the rotation matrices (B.4) for the plane adjust, x_0 and y_0 the coordinates of the non-tilted ellipse, σ_{sf} the scaling factor (B.12), and R_η is the rotation matrix for the ellipse tilt angle η. The yaw angle can then be computed using the calibrated readings $\mathbf{m}_C$. In Fig.B.6 is possible to see the difference between the raw and calibrated data with a reference value given by a second system.

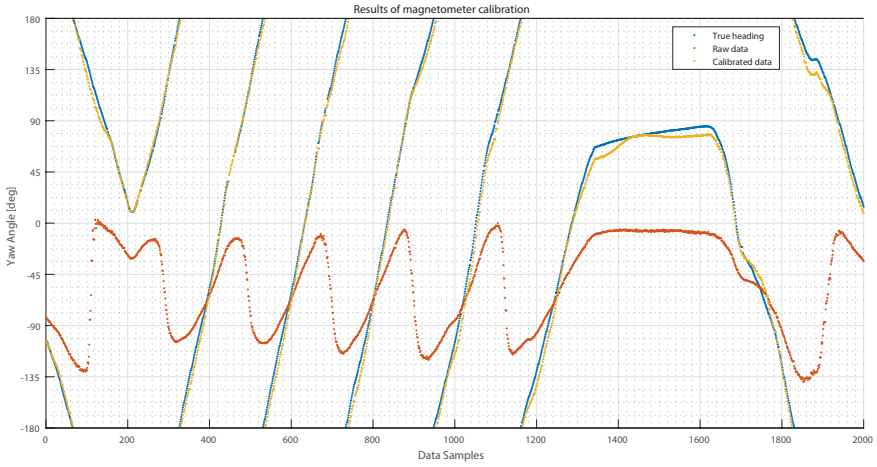

Figure B.6: Comparison of yaw angle computed from raw data, yaw computed from calibrated data, and reference given by a second more precise system.

Appendix C

Raspberry Pi 3 - Shield

The shield designed for the Raspberry Pi 3, serves several proposes. Being them, the protection of the Raspberry Pi from wrong connections, to facilitate the implementation and connections, and to remove the tangle of cables on top of the vehicle.

To protect the raspberry the shield should have two optocouplers, one for each PWM signal. A tension regulator was implemented since the actuation of both the motor and the servo require a 3.3V logic signal and the only source from the RC vehicle side is the battery.

Also, two LED's were implemented. One to indicate that the tractive system is on. And a second connected to a digital output of the Raspberry for debug and information.

The shield prototype made is presented in Fig.C.1. The PCB design is also presented in Fig.C.2 for identification of the pins number related to the schematic of Fig.C.3. The list of materials is:

- $\times$1 LDO (MCP1702-3302)
- $\times$2 Optocoupler (4n35)
- $\times$2 Capacitors (1μF)
- $\times$4 Resistors (560Ω)
- $\times$2 Resistors (1kΩ)
- $\times$1 Red LED
- $\times$1 Green LED
- $\times$1 Female Header (40$\times$2)
- $\times$2 Male Header (3$\times$1)
- $\times$1 Male Header (4$\times$1)

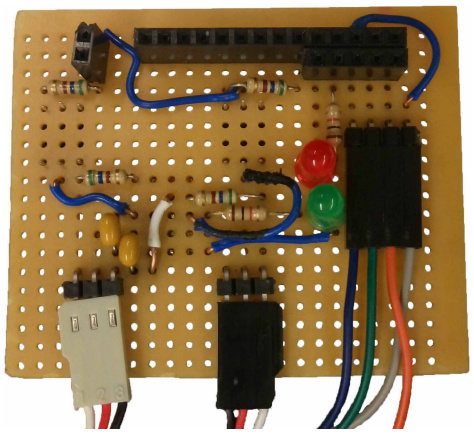

Figure C.1: Prototype shield.

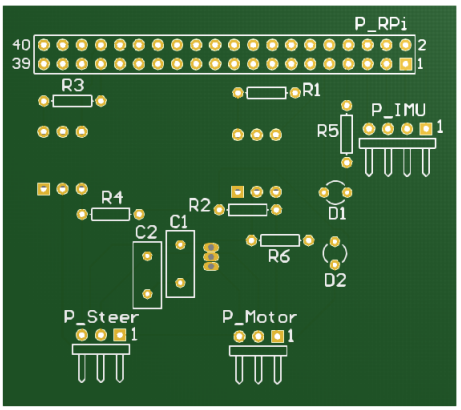

Figure C.2: PCB design of shield.

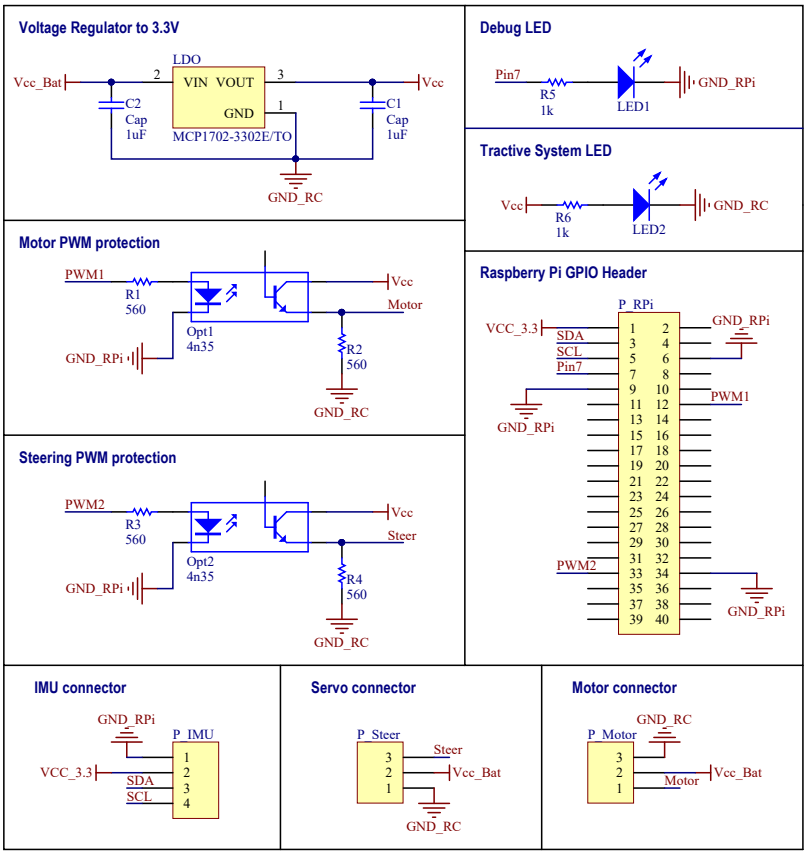

Figure C.3: Schematic of the shield.

The voltage regulator circuit presented in Fig.C.3 is the one proposed in the datasheet. The circuit for the optocoupler was design with a commitment between fast output response and low current consumption.

For the optocoupler emitter resistor was found that around 500Ωwas a good compromise between the fast response and a low current. It was used 560Ωresistor due to availability in the laboratory. The forward current (I_f) is then:

$$I_f = \frac{3.3V}{560\Omega} = 5.9mA \tag{C.1}$$

From the optocoupler datasheet is found that the current transfer ratio (CTR) is 100%, resulting in the same current at the collector (I_c) since:

$$I_c = CTR \times I_f \tag{C.2}$$

The output is supplied by a Vcc_r=3.3V from the regulator, and for the logic signal to have the same 3.3V the condition (C.3) must verify. This means the tension lost will be less or equal to zero.

$$Vcc_r - R_o I_c \leq 0 \tag{C.3}$$

Which results in a resistor output R_o value of at least 560Ω. This is expected due to the CTR value.